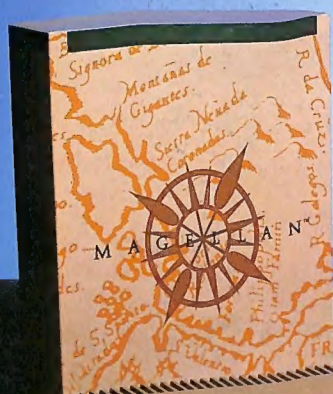
Gennaio/Febbraio 1989, Anno IV n. 1

La rivista per utenti di C-64/128 ed Amiga

COMMODORE GAZETTE

Intelligenza Artificiale:

MAGELLAN, UN SISTEMA ESPERTO PER L'AMIGA

Sped. in abb. post. Gr. III/70 – Lire 8.000

Programmare l'Amiga:

LE LISTE DELL'EXEC

L'AZTEC C DELLA MANX

I GADGET BOOLEANI

LE MACRO ASSEMBLY

Stampanti alla prova:

LE OKI 192 E 292 ELITE

Da Francoforte:

PARLANO I PROGRAMMATORI

Listati:

ECLISSI SUL C-128

Il mondo di GEOS:

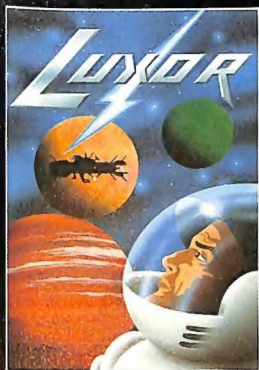
GEOS 2.0 E GEOCHART

I BOX DI DIALOGO

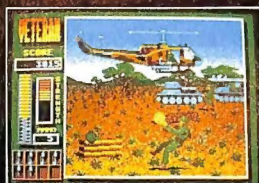
GEOS
NON HA PIÙ SEGRETI

Sei un utente di GEOS e ne sei entusiasta? Desideri scoprirne ogni dettaglio e impararne il funzionamento? Vuoi programmare in questo sistema operativo ad ambiente grafico? Allora questo volume della IHT ti è indispensabile. Scritta dagli stessi creatori di GEOS, questa guida svela tutti i segreti del sistema operativo e permette di creare stupende applicazioni per il C-64 e il C-128 dotate di interfaccia utente grafica, menu, icone, finestre, box di dialogo, fonti proporzionali, processi in multitasking, RAM disk, gestione dei file in overlay, controllo del mouse, driver di input e di stampa, e tante altre caratteristiche che fanno di GEOS un ambiente di lavoro e sviluppo davvero professionale.

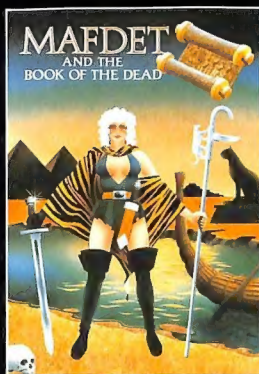
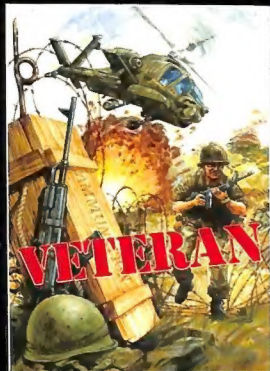
592 pagine, L. 64.000

I libri IHT sono disponibili nelle migliori librerie e computer shop. Per ordini diretti servirsi dell'apposito modulo pubblicato a pagina 127
IHT Gruppo Editoriale - Via Monte Napoleone, 9 - 20121 Milano - Tel. 02/794181-794122 - Fax 784021 - Telex 334261 IHT I
Distribuzione: Messaggerie Libri - Via Giulio Carcano, 32 - 20141 Milano - Tel. 02/8438141

Il Clan dei pirati dello spazio seguaci del dio del male OKIRA, hanno rapito la vostra adorata pricipessa DIORA e stanno ora tenendo l'impero in ansia per il riscatto . . .

Voi dovete perlustrare tre pianeti per trovare la vostra principessa, e ciascuno di essi sarà fortemente scortato da guerrieri di OKIRA estremamente addestrati. Trova il pianeta in cui DIORA viene trattenuta, distruggi i pirati, e una fata ti csaudira un unico desiderio.

Amore e magia ti aiuteranno ad uscire vittorioso — tu non devi fallire questa missione. Raccogli punti di forza per ricaricare le tue armi e i sistemi di sopravvivenza in quanto tu combatti per aprirti un varco tra i pianeti.

VETERAN

La guerra è quasi terminata. La tua fazione sembra aver perso tutto. Ma c'è solo un'ultima possibilità per salvare la battaglia.

Tu devi lottare per portare a termine con successo 3 missioni mortali, abbattendo le posizioni nemiche, finchè tu non raggiungerai il Quartiere Generale dei nemici. Fai esplodere i serbatoi, le piazzole delle armi, e abbatti i gruppi dei soldati nemici. Le tue provviste sono limitate, per questo tu devi raccogliere lancia razzi ricambio e munizioni per la tua pistola, lasciati dai soldati in fuga.

La storia inzia nell'antico Egitto. Il Dio del male, SETH, ha rubato il Libro dei Morti e lo ha nascosto nel profondo dell'Inferno.

Tu sei Mafdet, il gatto Dio della Vendetta, il quale e stato convocato da ISIS (Dio della Terra) per recuperare il Libro e restituirlo al propolo Egiziano.

Raccogli amuleti che ti niuteranno nel tuo cammino; trasformati da gatto a umano per superare alcuni ostacoli; combatti per apirti un varco tra labirinti e prigioni; soprattutto — TROVA IL LIBRO.

3 SUPEBI GIOCHI PER IL TUO ATARI ST O COMMODORE AMIGA

SUPERBI SONO ANCHE LA GRAFICA IL SUONO E LA GIOCABILITA

SOLO A LIT. 29.000 CADAUNO

Disponibile presso tutti i migliori grossisti o direttamente presso la

ITAL VIDEO SRL CASTENASO (BOLOGNA)

TELEFONO 051/784010

Articoli

FOTO PATRICIA LEEDS

GENNAIO/FEBBRAIO 1989
Anno IV, N.1

Rubriche

COMMODORE
GAZETTE

La rivista per utenti di C-64/128 ed Amiga

COMMODORE GAZETTE

Una pubblicazione

Direttore Responsabile
Massimiliano M. Lisa

Redazione
Luca Giachino (capo servizio redazione tecnica)
Mauro Gaffo (redattore)
Dario Greggio (redattore tecnico)
Nicolò Fontana Rava (servizi speciali)
Collaborazione Editoriale
Fabio Rossi, Alfredo Macchi, Marco Menichelli,
Franco Toldi, Sergio Fiorentini
Corrispondenti USA
William S. Freilich (sezione sviluppo)
Daniela D. Freilich (coordinamento generale)
Collaborazione Editoriale USA
Louis R. Wallace, Ervin Bobo,
Eugene P. Mortimore, Harriet Maybeck Tolly

Inviato speciale USA
Matthew Leeds

Segretaria di Redazione
Giovanna Varia

Impaginazione e Grafica
Antonio Gaviraghi (capo servizio)
Andrea De Michelis
Fotografia
A.&G.,
Patricia Leeds
Disegni
Marco Piazza, Giuseppe Festino

Direzione, Redazione, Amministrazione
IHT Gruppo Editoriale S.r.l.
Via Monte Napoleone, 9 - 20121 Milano

Fotocomposizione
IHT Gruppo Editoriale S.r.l. - Reparto grafica
Fotolito
C.S.T. S.r.l.
Via Melchiorre Gioia, 61 - 20124 Milano
Stampa
Eurograph S.p.A.
Via Oroboni, 38 - 20161 Milano

Distribuzione per l'Italia
Messaggerie Periodici S.p.A.
V.le Famagosta, 75 - 20142 Milano
Tel. 02/8467545

Distribuzione per l'estero
A.I.E. S.p.A.
Via Gadames, 89 - 20151 Milano
Tel. 02/3012200

Pubblicità
IHT Gruppo Editoriale S.r.l.
Via Monte Napoleone, 9 - 20121 Milano
Ufficio Pubblicitario IHT
Italia ed Estero 02/794181-799492-792612-794122
Fax 02/784021 - Telex 334261 IHT I

Servizio Abbonamenti

IHT Gruppo Editoriale
Servizio Abbonati
Via Monte Napoleone, 9 - 20121 Milano

SEGRETERIA ABBONAMENTI
Linea per registrazione abbonamenti
02/794181-799492-792612-794122

Commodore Gazette
Costo abbonamenti

Italia
12 numeri L. 96.000
24 numeri L. 192.000
36 numeri L. 288.000
Estero:
Europa L. 150.000 (10 numeri)
Americhe, Asia... L. 200.000 (10 numeri)

I versamenti devono essere indirizzati a:
IHT Gruppo Editoriale S.r.l.
Via Monte Napoleone, 9 - 20121 Milano
mediante emissione di assegno bancario o vaglia postale

Arretrati
Ogni numero arretrato: L. 16.000 (sped. compresa)

Autorizzazione alla Pubblicazione
Tribunale di Milano n. 623 del 21/12/85
Periodico Mensile - Sped. in abb. post. gr. III/70
ISSN: 0394-6991

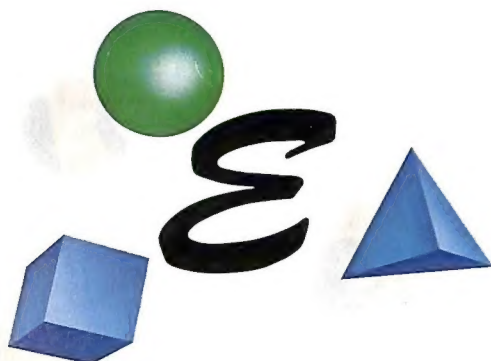
La IHT Gruppo Editoriale è iscritta nel Registro Nazionale della Stampa al n. 2148 vol. 22 foglio 377 in data 5/6/1987

Commodore Gazette è una pubblicazione IHT Gruppo Editoriale
Copyright © 1989 by IHT Gruppo Editoriale S.r.l.
Tutti i diritti riservati.
Nessuna parte della rivista può essere in alcun modo riprodotta senza autorizzazione scritta della IHT Gruppo Editoriale. Manoscritti e foto originali, anche se non pubblicati, non si restituiscono.
I contributi editoriali (di qualunque forma), anche se non utilizzati, non si restituiscono.
Non si assume alcuna responsabilità per eventuali errori od omissioni di qualsiasi tipo.

Commodore Gazette è un periodico indipendente non connesso in alcun modo con la Commodore Business Machines Inc. né con la Commodore Italiana S.p.A. PET, CBM, Vic-20, C-64, C-128, Amiga... sono marchi protetti della Commodore Business Machines. Nomi e marchi protetti sono citati senza indicare i relativi brevetti. Macintosh è un marchio della Apple, IBM PC AT, XT... sono marchi protetti dalla International Business Machines.

Associato alla U.S.P.I. (Unione Stampa Periodica Italiana)

USPI

NOTE EDITORIALI

In corrispondenza delle vacanze il Governo italiano ha l'abitudine di ricorrere all'odiosa, aberrante e oppressiva forma del decreto-legge su argomenti particolarmente spinosi. Forse nella convinzione che gli italiani in prossimità delle ferie siano più tolleranti dinanzi a provvedimenti cervellotici.

Prima dell'estate è stata la volta del ministro Ferri, che da oscuro peone della politica italiana è diventato un personaggio notissimo a tutti grazie alla sua crociata contro la velocità. Dimenticandosi che nel nostro Paese esistono problemi ben più gravi da risolvere, ha tentato di dare origine alla razza degli automobilisti-lumaca. Tanto a lui cosa importa? A spese dei contribuenti può tranquillamente turbinare ovunque a 200 all'ora con le scorte a sirene spiegate oppure sfrecciare nei cieli con jet ed elicotteri.

Tra Natale e Capodanno è stata la volta del cossiddetto "decretone fiscale", che tra le tante assurdità ne contiene una dalle proporzioni ciclopiche. A solo un anno dall'abolizione dell'IVA del 2% sui libri, dal 1° gennaio '89 è stata reintrodotta l'IVA su libri, quotidiani e periodici con l'aliquota maggiorata al 4%. Un'assurda tassa sulla cultura e l'informazione scritta, un balzello profondamente antidemocratico.

Si tratta di un prezzo amaro per l'editoria, perché l'IVA porta con sé anche un pesante fardello gestionale che si rifletterà in misura ben maggiore del 4% sui prezzi al pubblico. E prezzi di copertina maggiori vogliono dire meno lettori. E meno lettori vogliono dire peggiore andamento di un settore come quello dell'editoria già messo in crisi dal fatto che in Italia si legge poco.

Ci è stato raccontato che si tratta di una norma in favore di un'armonizzazione con quanto avviene negli altri Paesi europei. Non è vero. Proprio a livello CEE si sta decisamente profilando l'orientamento a esentare dall'IVA giornali, periodici e libri. Tant'è che la stessa CEE ha già compreso i prodotti editoriali tra quelli che dovrebbero essere allineati all'aliquota zero.

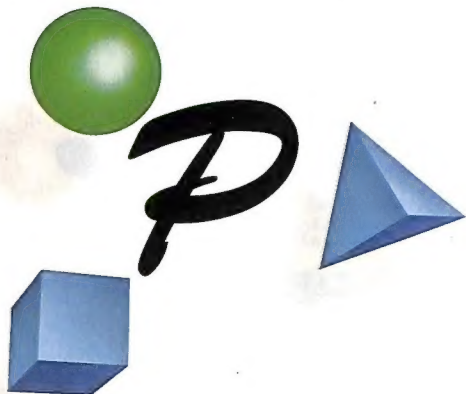
Ci è stato raccontato che le casse dello Stato hanno urgente bisogno di entrate. In questo contesto, il gettito derivante dall'IVA sull'editoria è talmente scarso da risultare irrilevante e quindi inutile.

Quello che non ci è stato raccontato con molta chiarezza è che quasi contemporaneamente all'introduzione della tassa sulla cultura il potere politico ha ritenuto opportuno proporre l'aumento del finanziamento pubblico ai partiti. L'ennesima contraddizione di un Paese in cui i miliardi spariscono dalle casse dello Stato per confluire nelle speculazioni sui terremotati, nelle carceri e nelle lenzuola d'oro.

Al momento di andare in stampa con questo numero, il Parlamento non ha ancora adottato alcuna decisione in merito alla conversione in legge del decreto che impone la tassa sulla cultura. Nel frattempo sono emersi pareri contrari a questo balzello anche a livello politico, il più autorevole dei quali è stato espresso dal presidente del Senato, Giovanni Spadolini. La nostra speranza è quella che il buon senso, alla fine, prevalga. La cultura non ha prezzo e non può essere monetizzata con nessuna tassa.

ML

LA POSTA DELLA GAZETTE

LA VOCE DEI NOSTRI LETTORI

UN CLUB DI PROGRAMMATORI

Sono un vostro lettore di Magliano dei Marsi e acquisto quasi sempre la vostra rivista. Posso dire di essere soddisfatto di tutto meno che della posta, perché pubblicate sì lettere interessanti, che magari aiutano un po' tutti, però è impossibile che non vi arrivi qualche missiva ogni tanto che parla di fondare club, ma badate bene, club di programmazione e non i soliti scambio/vendita giochi. Ebbene, capitandovi sicuramente qualcuna di tali lettere, voi non le pubblicate. Naturalmente voi direte: «Sì, questo dice 'ste cose perché vuole che gli pubblichiamo la lettera». Ebbene sì, lo ammetto, ma questo è solo uno dei motivi.

Ad esempio potrei dirvi che la mancanza di software di origine italiana e la pacchia dei pirati sono dovute più alle riviste che alle persone. Voi direte: «Perché?». Ebbene, il motivo è semplice: nel primo caso, le riviste che non pubblicano questo tipo di annunci disorientano i programmatori che non riescono a trovare quel qualcosa che li porti a produrre giochi. E poi, capirai, se il loro gioco lo commercializzano, lo danno a case straniere che poi lo passano per un prodotto inglese o americano o di qualche altra parte, scrivendo del suo realizzatore solo un piccolo nomignolo nascosto dai loro loghi. Il secondo caso invece è ancora più semplice: voi la fareste una legge che vieti la copia o la riproduzione di qualcosa che in realtà non c'è?

E, tanto per essere chiari, fare una legge che vieti la riproduzione di software è impossibile, perché non ci sono ditte che richiedono tale protezione (quelle inglesi lo fanno solo in casa, e non in Italia) e per questo i pirati da edicola non smettono di scrivere schifezze sui lavori di altri. In alternativa, io direi a voi di pubblicare ogni tanto una lettera tipo la mia, e agli altri (al popolo) di pensare a fondare qualcosa, perché in Italia si possono fare ottime produzioni in quanto i programmatori non mancano (vedi quella certa Simul... not pubblicità).

Quindi apritevi, e comunque se volete potete sempre scriverci inviando qualche vostro lavoro, o più semplicemente dimostrandoci che sapete fare qualcosa in LM per C-64, Amiga, Atari ST, IBM, Archimedes. Noi contatteremo le case di software italiane (sapete, ce ne sono addirittura un paio) e in seguito potremmo produrre da noi il software facendoci un nome. Per entrare a far parte del nostro club non servono soldi ma soltanto la conoscenza del linguaggio macchina e molta serietà in quello che fate.

Massimiliano Salustri
Via S. Maria
67062 Magliano dei Marsi (AQ)
(tel. 0863/517885)

Pubblichiamo volentieri la sua lettera, e le facciamo i migliori auguri per il club. Ma non riusciamo a capire per quale recondito motivo avremmo dovuto cestinare

Indirizzate tutta la corrispondenza per la rivista a:

COMMODORE GAZETTE
La posta della Gazette
Via Monte Napoleone, 9
20121 Milano

Preghiamo i lettori di essere concisi e concreti, per darci modo di rispondere al più grande numero possibile di lettere. La redazione si riserva comunque il diritto di sintetizzare le lettere troppo lunghe.

proprio le lettere dei programmatori italiani che ci scrivono per proporre la fondazione di club.

INFORMATICA GIUDIZIARIA

Sono un giovane vincitore del concorso per Uditore Giudiziario e, in vista del prossimo inizio della mia professione, mi sono equipaggiato con un sistema computerizzato per poter affrontare in chiave "moderna" l'ingente lavoro di scrittura a cui è chiamato il magistrato.

Il sistema a mia disposizione si compone dei seguenti elementi: Amiga 500, monitor a colori Philips CM8833, stampante Citizen MSP-10E.

Ho provato, inoltre, a casa di un amico, il word processor *LPD Writer* che mi ha molto favorevolmente impressionato inducendomi a programmare il suo acquisto. Nonostante questo, però, ho incontrato un grossissimo problema che mi preme particolarmente risolvere.

Il lavoro di scrittura, nel mondo giudiziario e del diritto in generale (mi riferisco non solo ai magistrati, ma anche ad avvocati, notai, commercialisti ecc.) il più delle volte dev'essere svolto su carta da bollo, o uso bollo, italiana. Quest'ultima, però, com'è ben noto, utilizza una spaziatura tra i righi realizzata con il sistema metrico decimale (per la precisione la distanza tra i due righi è di dieci millimetri) e tale fatto comporta, almeno per me, che non sono esperto di computer e programmazione, la materiale impossibilità di stampare sulla carta in oggetto (in modo tale, ovviamente, da rispettare le spaziature tra i righi) documenti elaborati con il word processor suddetto.

Umberto Antico
Via A. Sogliano, 19
80141 Napoli

Può rivolgersi alla Cloanto, una software house di Udine che produce un word processor italiano, C1-Text, *in grado di risolvere il suo problema. L'indirizzo è* Cloanto Italia, *Via G. B. Bison 24, 33100 Udine (tel. 0432/46612).*

SE LA CPU È LA STESSA...

Sono un felice possessore di un A2000. Spesso, pensando al processore di questa meravigliosa macchina, mi sono posto la seguente domanda: l'Apple Macintosh, l'Atari ST e il Commodore Amiga hanno tutti e tre lo stesso microprocessore, il grandioso Motorola 68000. Perché allora i programmi di una macchina non possono girare sull'altra? Quali sono le differenze sostanziali? Se i programmi di ognuno funzionassero sugli altri, si avrebbe uno strumento per contrastare i PC perché si avrebbe un'altra gamma di compatibili.

Guido Dicadani
Corsico (MI)

La sola presenza dello stesso microprocessore non può garantire la compatibilità di programmi scritti per macchine diverse perché il microprocessore non è l'unico elemento che caratterizza un computer. Esistono anche il sistema operativo, e le risorse hardware. Per esempio, il Macintosh, a differenza dell'Amiga, non possiede un sistema operativo multitasking, e la grafica è monocromatica. Sono differenze sufficienti per impedire a un programma di agire in entrambi gli ambienti, dal momento che per ricevere dati in input e generare dati in output deve impiegare le risorse del sistema per il quale è stato programmato.

L'unica possibilità di avere un programma "universale" è di non fargli utilizzare nessuna risorsa della macchina. In altre parole, non dovrebbe effettuare nessuna chiamata al sistema operativo, né prendere in considerazione registri diversi da quelli della CPU: un programma di questo tipo potrebbe girare virtualmente su qualsiasi computer dotato della stessa CPU, ma a cosa potrebbe servire? Se in ambito MS-DOS le cose vanno diversamente, è solo perché queste macchine – dal punto di vista delle applicazioni – sono praticamente identiche, tanto quanto sono invece diversi fra loro il Macintosh, l'Atari ST e l'Amiga.

MEZZO MEGA PER IL C-64

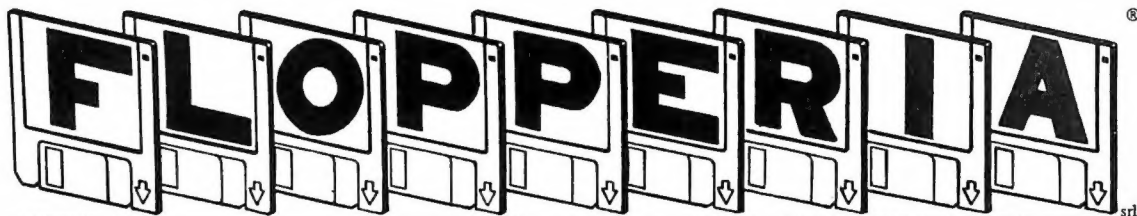
Ho un "piccolo" C-64 a cui sono molto affezionato e al quale affianco tutte le periferiche e le espansioni che mi posso permettere (con i soldi che ho speso avrei potuto acquistare tre Amiga 500!). A proposito di espansioni, vorrei rivelare un piccolo trucco utile ai possessori dell'espansione 1764 (256K) per il C-64. Questa espansione si può facilmente portare a mezzo Mega (512K) come la 1750 per il C-128 tramite la semplice aggiunta di otto chip di RAM dinamica 4256-15NL; infatti la Commodore ha utilizzato, com'è logico, la stessa scheda per entrambe le espansioni. E ora passo alle consuete domande.

Nella recensione di *BeckerBASIC* (n. 4/88 di *Commodore Gazette*), nelle ultime righe fate trapelare il nome di un interprete Basic, il *Basic Lightning* che a quanto pare dovrebbe essere piuttosto potente (strutture di controllo alla Pascal, procedure con passaggio di parametri ecc.), dove si può trovare in Italia questo programma? Chi lo distribuisce?

Paolo Carminati
San Mauro (TN)

Il Basic Lightning *non è un programma recentissimo, ma è tuttora disponibile in Italia. Lo si può trovare presso la Lago snc (Via Napoleona, 16 – 22100 Como – tel. 031/300174) e costa L. 39.000.* ■

FLOPPERIA®

srl

presenta

in esclusiva per l'Italia

la **miglior** cartuccia mai prodotta

per Commodore 64 e 128

MK V

Mk V' non solo è la miglior cartridge per effettuare copie di sicurezza del proprio software, ma è anche il più efficace velocizzatore nastro/disco e la più versatile cartuccia di utility esistente.

La sua peculiarità più innovativa è data dal suo **microprocessore interno**, appositamente studiato per sovrapporsi a quello del computer, ed assolutamente invisibile al sistema.

Ad esempio, mentre le altre cartucce si fermano ad un banale "Sprite Killer" per facilitare i giochi, **Mk V' è in grado di trovare automaticamente le "Poke" necessarie per le vite infinite** di qualsiasi programma presente e futuro, senza attendere che siano pubblicate dalle riviste o che qualche smanettone studi il programma. Ora anche voi potete produrre giochi **"trainer"**, senza alcuna conoscenza di linguaggio macchina! Inoltre **sprotegge e porta da nastro a disco (e viceversa) qualsiasi programma protetto, anche in multiload** (con i parametri in dotazione); può trasferire molti programmi e files dal formato 5"1/4 al nuovo **1581 da 3"1/2;** velocizza il nastro 5-6 volte oppure 8-10 volte, con velocità selezionabile; velocizza il disk drive come se fosse parallelo **(2 velocità: 202 blocchi in 9 secondi oppure in 6!)**, ed è sempre efficace, anche con i programmi che disabilitano i fastload normali. Mk V' incorpora un vero e proprio editor di schermo, per poter cambiare più facilmente e velocemente le scritte nelle schermate o nei programmi; **funziona da interfaccia parallela**, per collegare una qualsiasi stampante standard Centronics al C64/128 e di usarla all'interno di qualsiasi programma, anche grafico; stampa o salva in qualsiasi momento la schermata o gli sprites di un gioco, per alterarli a piacimento. Aggiunge nuovi comandi al Basic, monitor L/M e disk, crea serie di immagini in sequenza su nastro, e tantissime altre cose ancora.

Per Commodore 64 e 128 (in modo 64), con qualsiasi registratore o disk drive, originali o compatibili.

In offerta a solo £. 115.000

(IVA e spedizione compresa!!!)

Viale Monte Nero, 31
20135 Milano

Tel. (02) 55.18.04.84
Fax (02) 55.18.81.05

C-64 e C-128 sono marchi registrati della Commodore Business Machines. Pagina a cura di IdeaMatix Srl.

SOFTWARE GALLERY

UNA GUIDA PER ORIENTARSI NEL MONDO DEL SOFTWARE

BATTLE CHESS

Computer: Amiga
Supporto: Disco
Prezzo: L. 59.000
Produzione: Interplay (Electronic Arts)
Distribuzione: C.T.O. (Via Piemonte 7/F, 40069 Zola Predosa – 051/753133)

Ogni computer ha gli scacchi che si merita: sullo ZX81 era praticamente impossibile perdere, sul C-64 alcuni programmi danno problemi anche a giocatori esperti e sull'Archimedes perfino Fischer suderebbe non poco per riuscire ad aggiudicarsi il match. Fra queste due ultime macchine sta l'Amiga, un computer dotato di una veloce CPU, ma anche di capacità grafiche e sonore quasi imbattibili. Era inevitabile che i programmi scacchistici per l'Amiga fossero unici, e questo *Battle Chess* è infatti completamente diverso da qualsiasi altro programma di scacchi di questo pianeta.

Le sue caratteristiche di gioco sono notevoli: controllate da menu a discesa, abbiamo a nostra disposizione opzioni che faranno la gioia di ogni appassionato del nobile gioco. È possibile giocare contro il computer, far giocare la macchina contro se stessa (è il primo caso di schizofrenia a 16 bit) o giocare con un'altra persona. Se poi l'altra persona in questione abita dall'altra parte del globo, non c'è problema: i progettisti della Interplay hanno pensato bene d'includere la possibilità di collegarsi via modem con un altro Amiga e di poter quindi ingaggiare sfide con i campioni di tutto il mondo... sfide tanto affascinanti quanto dispendiose, se si pensa alla bolletta telefonica.

Giocando contro il computer, possiamo scegliere tra dieci livelli di difficoltà, con l'ultimo dei quali viene sfruttata tutta la potenza di calcolo del nostro Amiga. Quello che cambia da un livello all'altro è il tempo a disposizione della macchina per pensare alla mossa successiva, che va da un secondo nel livello per principianti a ben 22 inquietantissimi minuti nel livello dieci, riservato ai veri esperti.

Una volta selezionato il livello e arrivati sulla scacchiera, possiamo scegliere il colore fra rosso e blu, caricare una partita lasciata a metà precedentemente o iniziarne una nuova, oppure sfruttare – a partita iniziata – il menu Move.

Questo menu risulta molto utile ai giocatori più confusionari e impazienti, e comprende quattro utilissime opzioni. La prima, Force Move, interrompe i ragionamenti del computer obbligandolo a compiere la miglior mossa trovata sino a quel momento. Take Back riporta la partita indietro di una mossa, e selezionando ripetu-

SCHEDA CRITICA

INSUFFICIENTE
Un pessimo prodotto che non merita nessuna considerazione.

MEDIOCRE
Il programma ha alcuni difetti di fondo, anche se nel complesso raggiunge quasi la sufficienza.

DISCRETO
Un prodotto accettabile, ma non aspettatevi grandissime emozioni.

BUONO
Raccomandato vivamente: tra i migliori programmi della sua categoria.

OTTIMO
Eccezionale! Fino ad oggi non si era mai visto niente del genere.

tamente questa opzione possiamo tornare addirittura alla situazione iniziale. Replay ha esattamente la funzione opposta, dal momento che porta avanti di una mossa una partita "arretrata" con l'opzione precedente. L'ultima possibilità, Suggest Move, scambia momentaneamente i ruoli dei giocatori volgendo a nostro vantaggio la capacità tattica del computer, che compie per noi la mossa migliore.

Esiste ancora un'ultima possibilità, contenuta in un diverso menu, che permette di collocare a piacere i pezzi sulla scacchiera: si può utilizzarla per dare un handicap a un giocatore (per esempio, facendolo giocare senza gli alfieri) o per studiare i problemi scacchistici presentati dalle riviste.

Certamente questa marea di possibilità è molto interessante, ma tutto sommato anche alcuni programmi per C-64 le contemplavano (*Colossus Chess 7*, per esempio). Cos'ha di tanto speciale, allora, *Battle Chess*?

Ebbene, questi scacchi della Interplay possono essere giocati sia in due dimensioni, come la maggior parte dei programmi scacchistici, sia in tre, e in questo caso il gioco si trasforma in uno spettacolo senza precedenti.

Sulla scacchiera, che può essere scelta in modo che sembri di marmo oppure con caselle monocromatiche, i pezzi appaiono vivi, disegnati e animati al meglio delle capacità dell'Amiga. Quando viene selezionata una mossa, il pezzo interessato cammina attraverso la scacchiera accompagnato dai rumori (in stereo, è ovvio!) tipici del suo personaggio. Un pezzo dotato di armatura ha un passo pesante e metallico, mentre i provocanti movimenti della Regina sono accompagnati da un fruscio di seta. Persino la Torre prende una forma antropomorfa quando è il momento di passare all'azione, e si trasforma in un terrorizzante gigante di pietra dagli occhi fiammeggianti che avanza implacabile facendo tremare il terreno sotto i suoi granitici piedoni. I quattro megabyte impiegati per l'animazione non vengono però spesi tutti nelle sequenze di movimento... si fanno notare molto di più in un'altra occasione: il combattimento!

Quando due pezzi si trovano sulla stessa casella, l'inevitabile vittoria dell'ultimo arrivato viene presentata nella maniera più spettacolare possibile, in un vero e proprio scontro all'ultimo sangue che ricorda molto la sequenza degli scacchi olografici di *Guerre stellari*. I brevi combattimenti variano a seconda dei pezzi, con risvolti spesso umoristici che lasciano a bocca aperta.

Se i Pedoni si limitano a decapitare il pezzo nemico, la potente Torre schiaccia sadicamente con un pugno di pietra i suoi avversari e gli Alfieri maneggiano come samurai giapponesi un'asta magica con cui lanciano raggi mortali. L'eccezione è ovviamente rappresentata dal Re, che osserva impassibile gli sforzi della sua vittima per poi freddarla con una Magnum 45 che estrae dal mantello di ermellino!

Battle Chess è senza dubbio un programma divertente da giocare ma soprattutto da vedere. Il suo tallone d'achille è però rappresentato proprio da quei combattimenti che ne sono anche il vanto. Ogni mossa sulla scacchiera tridimensionale richiede il caricamento delle sequenze di suono e animazione relative al pezzo coinvolto, e un combattimento richiede un altro breve caricamento, al tempo del quale va poi aggiunto quello della battaglia vera e propria. Visti sotto quest'ottica, gli scontri fra i pezzi di *Battle Chess* diventano presto una noia, dal momento che pur essendo fantastici mantengono sempre la stessa struttura.

Dopo le prime due o tre partite giocate fra le urla dei pezzi colpiti e il clangore delle spade che si incrociano, la grafica e il sonoro di questo programma diventano soprattutto un impiccio, che molti decideranno di eliminare ricorrendo alla scacchiera bidimensionale. Soltanto così *Battle Chess* smette di essere un raffinato giocattolo e si trasforma in un ottimo programma di scacchi, la cui libreria di 30 mila aperture darà più di un grattacapo a tutti coloro che hanno deciso d'imparare con l'Amiga il gioco più coinvolgente del mondo.

F.R.

MICROPROSE SOCCER

Computer: C-64/128
Supporto: Cassetta/Disco
Prezzo: L. 39.000/49.000
Produzione: Sensible Software (Microprose)
Distribuzione: Leader (Via Mazzini 15, 25020 Casciago – 0332/212255)

Ecco che finalmente anche la Microprose si lancia nel vasto e colorato universo dei giochi di simulazione sportiva, e, con l'uscita di *Microprose Soccer*, continua a mantenere fede a quella tradizione di qualità che la software house è riuscita a costruirsi negli ultimi anni.

Dalle menti dei programmatori della Microprose sono nati giochi come *Airborne Ranger*, *The President is Missing*, *Stealth Fighter*, che hanno ottenuto un buon successo di pubblico e di critica. In linea con questa impostazione e con lo slogan della casa: "easy to learn, difficult to master", *Microprose Soccer* è un'ottima simulazione calcistica, probabilmente una delle migliori viste fino a oggi. Con la sua ottima prospettiva dall'alto e una grafica degna dei migliori arcade, questo gioco unisce rapidità e semplicità d'azione a tutta una serie di opzioni che rendono ogni partita molto realistica, senza appesantire troppo l'azione.

Il programma prevede diverse possibilità di gioco, dalla partita ufficiale con undici giocatori contro undici, fino a un interessante sei contro sei indoor (il cosiddetto calcetto). È ammesso il gioco a due, ma se si preferisce ci si può sempre confrontare con il computer. È addirittura possibile organizzare una vera e propria coppa del mondo (con tanto di gironi eliminatori e classifiche) che prevede la partecipazione di 24 nazioni, ognuna dotata di un proprio stile di gioco e di una propria forza. Ciò permette di organizzare tornei di gruppo che possono essere salvati e ripresi nei giorni seguenti. Anche per il giocatore solitario, comunque, esiste una "international challenge", nella quale si affrontano squadre via via più forti controllate dal computer. A tutto ciò vanno aggiunte varie opzioni che permettono di controllare la durata delle partite, i livelli di difficoltà, le condizioni atmosferiche (è anche possibile giocare sotto la pioggia, e la differenza si sente!).

Per quanto riguarda l'azione vera e propria, bisogna dire innanzitutto che il controllo è piuttosto agevole e non smentisce lo slogan della Microprose. Dopo poche partite si riesce a ottenere il completo controllo dell'azione e degli uomini in campo. La prospettiva, come già detto, prevede solo la vista dall'alto e ricorda un po' – anche nel modo di controllare i giocatori – il famoso gioco da bar *World Cup*.

Il giocatore controlla l'uomo che lampeggia o che è in possesso di palla e lo muove nelle otto direzioni con il joystick; se si è in possesso di palla il tasto Fire serve per tirare (la forza del tiro è proporzionale al tempo di pressione), altrimenti serve per entrare in scivolata in un tackle. Il tiro a effetto si ottiene spostando il joystick dopo aver tirato. Sullo schermo, naturalmente, non appare tutto il campo ma solo il settore dove si muove il giocatore in possesso della palla; in ogni caso è un settore sufficientemente ampio da consentire una buona visione d'insieme del gioco. Se un avversario si avvicina troppo alla nostra area un segnale acustico ci avverte che il controllo del joystick passa al portiere, il che ci permette di compiere spettacolari parate e di difendere la porta con tempestiva efficienza.

Si tratta di un gioco semplice e completo: dribbling, passaggi, tiri a effetto, rovesciate, parate... il tutto accompagnato da un'ottima grafica e da un sonoro ben realizzato. Se teniamo conto anche della semplicità dei comandi e della varietà di possibilità offerta al giocatore, non è difficile prevedere che diventerà un pezzo molto richiesto dagli appassionati di calcio o di giochi sportivi su computer in generale. I mondiali non sono lontani! **N.F.R.**

UMS

Computer: Amiga
Supporto: Disco
Prezzo: L. 45.000
Produzione: Rainbird
Disponibile presso: Lago (Via Napoleona 16, 22100 Como – 031/300174)

Chi fra i nostri lettori ha giocato almeno una volta con un war game conoscerà sicuramente quale sensazione di potenza si prova quando ci si trova di fronte a una mappa a esagoni. Le nostre armate, rappresentate da migliaia di segnalini di cartone, sono pronte a eseguire anche i più folli ordini; e riuscire a sconfiggere l'avversario in uno di questi giochi dà una soddisfazione veramente notevole. Sulle enormi mappe di cartone dei war game è possibile ricreare tutte le grandi battaglie del passato, misurandosi così contro grandi avversari come Napoleone o Rommel, o si possono ipotizzare scenari di guerra futuri (sperando che non si avverino mai). Soprattutto negli Stati Uniti, la passione per i war game ha dato vita a vasti circoli di appassionati che vivono il loro tempo libero nei panni di Alessandro Magno o di Hitler, studiando a volte per anni la realizzazione di un'utopistica "simulazione perfetta".

Dal momento che le variabili in gioco in una simulazione aumentano proporzionalmente alla sua complessità, affidare al computer i calcoli relativi alle azioni è stato un passo ovvio sia per gli appassionati sia per le ditte specializzate. Oltretutto così si risolveva anche il problema pratico di dover conservare da qualche parte mappe di molti metri quadrati, ricoperte da migliaia di leggeri e minuscoli segnalini di cartone. Aziende come la SSI o la SSG australiana hanno basato le loro fortune sulla creazione e la vendita di simulatori per home computer d'impressionante precisione.

L'unico problema di questi programmi, sino a qualche tempo fa, era dato dalle loro limitate possibilità. Per avere una simulazione accurata erano necessari programmi specifici per ogni battaglia. Visto anche il notevole costo di produzione, gli appassionati non erano particolarmente entusiasti di dover comperare una decina di dischi (sempre che fossero disponibili) per ricostruire un conflitto durato magari meno di una settimana.

Una delle persone meno entusiaste di questa situazione era sicuramente un certo Ezra Sidran che, preso il suo fido C-64, pensò bene di scrivere un programma polivalente in grado di ricostruire qualsiasi battaglia. Il progetto di Sidran si è protratto per anni e oggi, nonostante esista una versione di *UMS* anche per il C-64, la versione definitiva del suo *The Universal Military Simulator* è quella realizzata per l'Atari ST.

Come quasi tutti sanno, le differenze tecniche fra ST e Amiga sono minime sino a quando non si tirano in ballo le superiori capacità grafiche e sonore del mega-computer di casa Commodore, e la conversione di *UMS* per i computer Amiga è stata immediata e indolore.

The Universal Military Simulator, nella sua versione per l'Amiga, è contenuto in due dischi: il primo contiene cinque battaglie preconfezionate e l'utility per far girare una battaglia generica, e il secondo le utility per progettare la battaglia dei propri sogni.

Se carichiamo uno degli scenari dimostrativi, sul monitor appare una mappa tridimensionale composta di quadrati e costellata da un'infinità di simboli grafici e bandierine che riportano il nome del simbolo in questione. Attivando l'opzione relativa all'inizio della battaglia da una serie di sei eloquenti menu a discesa, appare una finestra sulla quale – una alla volta – sfilano tutte le nostre unità. "Premendo" con il mouse uno dei pulsantoni contenuti nella finestra si possono impartire gli ordini destinati a quella particolare unità, e i nostri comandi saranno eseguiti nei limiti del possibile durante il successivo turno di gioco. Una volta indirizzata anche l'ultima delle nostre unità, ha inizio il gioco (o meglio la battaglia), in cui si susseguono, alternati fra il nostro esercito e quello

avversario, una serie di "segmenti d'azione".

Mentre i nostri ordini vengono eseguiti, è possibile sfruttare le numerose opzioni fornite da *UMS*. La mappa può essere ruotata in ogni direzione, e una grossa finestra tridimensionale ci permette di zoomare su qualsiasi zona del terreno di battaglia, arrivando persino a osservare i movimenti di un singolo soldato! Possiamo richiamare altre finestre, grazie alle quali si studiano le statistiche relative alla battaglia, si cambia la rappresentazione grafica degli eserciti avversari (onde

d'urto, vettori, simboli o nomi), o si modificano i parametri di gioco.

Proprio questa possibilità d'interferire in qualsiasi momento nell'azione di gioco rappresenta l'asso nella manica di *The Universal Military Simulator*, che si rivela un prezioso strumento non solo per i fanatici dell'azione bellica ma anche per gli appassionati di storia, che possono così studiare cosa sarebbe successo, per esempio, se durante la battaglia di Waterloo Napoleone fosse riuscito a sferrare un attacco alle spalle delle linee nemiche. Impossibile, certamente, ma il bello delle guerre simulate è proprio questo.

Sempre limitandosi a modificare i parametri degli scenari forniti nella confezione, è possibile anche trasformare i fanti di Cromwell ad Hastings in panzergreniadern hitleriani o in elicotteri Apache, dando il via a situazioni improbabili che presentano spesso notevoli sorprese. Sapevate per esempio che un gruppo di cinque carri armati unito all'esercito di Dario non sarebbe stato sufficiente per fermare l'avanzata di Alessandro Magno ad Arbela?

The Universal Military Simulator non si limita comunque a permettere al giocatore improbabili modifiche degli scenari forniti nella confezione. Anzi, uno dei suoi punti di forza è proprio il disco che permette di creare la propria battaglia personale. Le utility contenute in questo disco ci permettono di ridisegnare sia la mappa sia le icone relative alle forze in campo, definendo nomi e caratteristiche di ogni elemento di gioco. Il risultato dei nostri sforzi creativi può essere poi salvato in un blocco unico su di un altro disco, e potrà in seguito essere giocato o modificato proprio come i cinque scenari realizzati dagli autori.

Con un attento uso delle utility a disposizione, *UMS* rende possibile anche la creazione di scenari fantascientifici o fantasy, con la simulazione di raggi laser e incantesimi, e battaglie ambientate in tetri castelli o negli stretti corridoi di una base spaziale.

Dopo un'attenta analisi del programma, possiamo dire che ci sono soltanto tre aspetti tralasciati dall'autore: *UMS* non tiene nel debito conto le condizioni atmosferiche (la Campagna di Russia si trasforma da tragedia in una gita aziendale), le armi ad ampio raggio d'azione, ma soprattutto la possibilità d'includere nell'azione unità navali, che avrebbero reso possibile simulare azioni famose come lo sbarco in Normandia.

In realtà, l'incredibile flessibilità di *UMS* permette di convertire parte del terreno in acqua, ma l'impossibilità di definire adeguatamente dei mezzi da sbarco o anche semplicemente navali, rappresenta una notevole limitazione. In un'apposita sezione, Sidran spiega d'aver lasciato perdere le armi nucleari perché avrebbero reso gli scenari troppo brevi, ma ciò non toglie che sia gli ordigni a fissione sia gli attacchi chimici sono purtroppo parte integrante della realtà militare del nostro tempo, e non ci sarebbe voluto molto per trasformare questo prodotto in un simulatore veramente completo.

Nonostante le piccole mancanze a cui abbiamo accennato, *The Universal Military Simulator* è senz'altro il massimo che un appassionato di war game possa trovare su computer. Inoltre, la possibilità di stampare su carta tutte le cartine, le situazioni e le statistiche fa di *UMS* un prodotto prezioso anche per gli chi ama le ricostruzioni storiche. Magari potrebbe essere un aiuto per decifrare le molte storie di guerra che ancor oggi rappresentano un enigma.

F.R.

WHO FRAMED ROGER RABBIT

Computer: C-64/128/Amiga
Versione: C-64/128
Supporto: Disco
Prezzo: L. 55.000/69.000
Produzione: Buena Vista Software
Disponibile presso: Lago (Via Napoleona 16, 22100 Como – 031/300174)

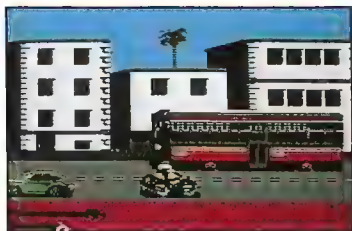
Chi non ha ancora sentito parlare di *Chi ha incastrato Roger Rabbit*, il film di Robert Zemeckis? Riprendendo il vecchio tema disneyiano della fusione tra personaggi reali e cartoni animati, il regista di *Ritorno al Futuro* ha ottenuto eccezionali consensi di pubblico e di critica. I suoi personaggi animati hanno riempito per mesi le copertine dei periodici di tutto il mondo e il film si appresta a polverizzare ogni record d'incasso. Questa volta, oltre ai fumetti e agli innumerevoli gadget basati sui personaggi e le situazioni del film, c'è anche un gioco dedicato a tutti i possessori di C-64.

Il programma è stato realizzato dalla Buena Vista Sofware, e ha come protagonista il simpatico coniglio "incastrato" nel film, che qui si ritrova ad affrontare alcune avventure in parte legate alla trama della produzione cinematografica.

Innanzitutto ci sono da fare alcune considerazioni riguardo all'idea e ai contenuti del gioco: è senz'altro da lodare la tempestività dei programmatori della casa americana e l'abilità dimostrata nel ricreare situazioni, ambienti e gag che trovano un certo riscontro nella pellicola. Purtroppo, però, i contenuti del gioco non raggiungono neanche lontanamente l'originalità della pellicola a cui si ispirano, secondo il classico meccanismo di nascondere la propria mediocrità dietro il paravento di un nome che è sulla bocca di tutti. Nonostante la discreta qualità della grafica e l'ambientazione inconsueta e divertente, la scarsa giocabilità di *Roger Rabbit* non ci permette assolutamente di definirlo un gioco riuscito.

Lo scopo del gioco, come quello del protagonista del film, è contrastare il malvagio giudice Doom nel suo tentativo di conquistare Cartoonia, la città dei cartoni animati, per poterla amministrare con pugno d'acciaio in guanto d'acciaio a suon di "salamoia" (la sola sostanza in grado di sciogliere, e quindi uccidere, i cartoni animati). Per ottenere il suo scopo, Roger deve superare tre prove che corrispondono alle

tre parti in cui si suddivide il gioco. Nella prima parte Roger si ritrova alla guida di "Benny the cab" un taxi/cartone dall'andatura strampalata. Insieme devono precedere il giudice Doom all'Ink & Paint club, un night per cartoni animati nel centro di Hollywood, evitando le altre auto e le chiazze di salamoia seminate lungo il percorso. Nelle istruzioni sono citati presunti bonus che dovrebbero trovarsi lungo il percorso, ma con comprensibile disappunto abbiamo constatato che nessuno degli oggetti descritti si trova sulla strada del nostro coniglio.

Superata questa fase ci si ritrova all'interno dell'Ink & Paint club dove si devono freneticamente recuperare tutti i menù sui tavoli, perché in uno di questi è

nascosto il testamento che impedirebbe a Doom di prendere possesso di Cartoonia. Nello svolgere questo compito bisogna essere più rapidi dei pinguini-camerieri che ripuliscono i tavolini, e fare attenzione a non far prendere a Roger i bicchieri di vino (chi ha visto il film sa che Roger diventa un po' nervoso quando beve!) e a non farsi acciuffare dal gorilla che ci sbatterebbe fuori dal locale.

Se Roger riesce a raccogliere tutti i menù e a uscire dal locale prima che l'orchestra finisca di suonare, anche la seconda parte del gioco è stata superata. Si riprende la corsa su Benny the cab (in realtà si tratta di una ripetizione del primo schermo), questa volta per raggiungere la fabbrica di gag. Nell'edificio, il compito di Roger è innanzitutto quello di usare le gag disseminate sul terreno per sbarazzarsi delle iene del giudice Doom facendole ridere a morte, e infine affrontare il giudice in persona nello scontro finale per assicurare la salvezza a Cartoonia.

Anche se la trama e le ambientazioni possono sembrare divertenti, in realtà il gioco non è all'altezza della cornice in cui si svolge. Il compito del giocatore è molto ripetitivo e la mancanza di varietà e fantasia toglie la voglia di riprendere in mano il joystick dopo poche partite. A queste note negative si va ad aggiungere la necessità di ricaricare il programma dopo ogni partita, operazione oltretutto lenta a causa dei limiti del C-64 e della lunghezza del programma.

Una notazione curiosa per concludere. Ricordate che Roger Rabbit è ricercato per l'omicidio di Marvin, proprietario di Cartoonia e re degli scherzi? Ebbene, un divertente librettino allegato al programma (che contiene anche le due pagine d'istruzioni) riproduce l'immaginario catalogo della "fabbrica di scherzi" di Marvin, estate 1947. Sono illustrati oggetti e gag di ogni tipo, dal guantone a molla al chewing-gum "piccante", dal sigaro che esplode al pulsante che dà la scossa. Ma si trovano anche oggetti più stravaganti, come la pozione per rimpicciolire i cartoni, le pillole che fanno camminare all'indietro e il tappeto magico garantito per 12 lavaggi.

Qualche ordinazione verrebbe voglia di farla ma, per quanto riguarda il gioco, è consigliabile spendere altre 8 mila lire per rivedere il film piuttosto che impegnare una somma molto maggiore per un prodotto che di buono ha soltanto un nome... che certamente non merita.

N.F.R.

NEUROMANCER

Computer: C-64/128
Supporto: Disco
Prezzo: L. 30.000
Produzione: Interplay (Electronic Arts)
Distribuzione: C.T.O. (Via Piemonte 7/F, 40069 Zola Predosa – 051/753133)

Se il 1988 è stato l'anno dei giochi di ruolo, l'inizio del 1989 è stato caratterizzato da una nuova tendenza nella produzione del software d'intrattenimento: le avventure "graphic novel". A differenza delle normali avventure, caratterizzate da una grande quantità di testo e da input che avvengono principalmente da tastiera, i graphic novel affidano le loro descrizioni a una grande finestra grafica, affiancata da un pannello di icone e speciali finestre destinate ai messaggi.

Ma le differenze che corrono fra un graphic novel e una tradizionale avventura non si fermano qui: mentre in un programma come *The Pawn*, o anche nel graficissimo *Shadowgate*, il giocatore decideva con le sue azioni lo svolgimento della trama, in un graphic novel diventa solo uno dei tanti personaggi di un mondo che vive autonomamente. In un graphic novel, grazie alla rappresentazione dinamica che viene data del mondo in cui si svolge il gioco, niente e nessuno si ferma ad aspettare le decisioni del giocatore.

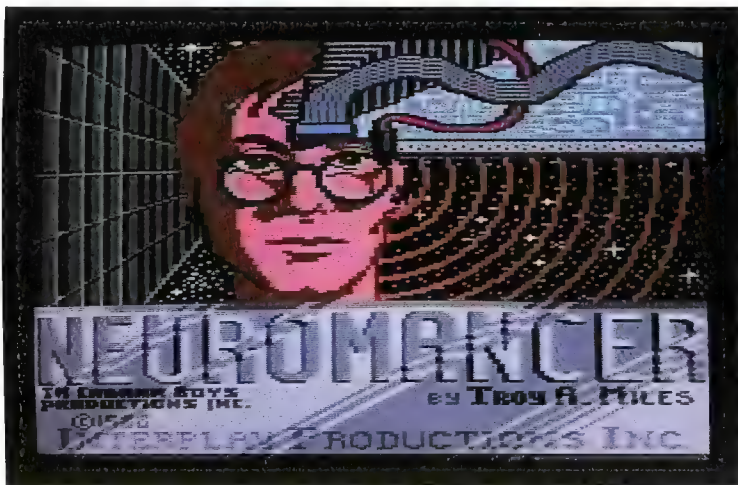
Questa particolarità, che può sembrare solo una fonte di frustrazione, permette invece al giocatore di entrare in un ambiente molto realistico, nel quale gli avvenimenti previsti dai programmatori del gioco avvengono con un ritmo tale da coinvolgere lo spettatore/giocatore come in un film. Proprio per questo, le trame dei graphic novel vengono spesso tratte da romanzi o film, cercando di mantenerne intatti ritmo, suspence e colore.

Dai tempi di *Labyrinth* della Lucasfilm, che si può considerare il primo graphic novel, i progressi nell'arte della programmazione hanno portato a vicende complesse e affascinanti come quelle di *Zak McKracken and the Alien Mindbenders* o questo *Neuromancer*, tratto da un romanzo di William Gibson.

Gibson è uno scrittore che nei Paesi anglosassoni è diventato una sorta di Umberto Eco fantascientifico. I suoi romanzi sono oggetto di veri e propri assalti alle librerie, e sono tanto apprezzati anche dai critici da aver segnato la nascita di un nuovo filone: il genere "cyberpunk" (che alcuni chiamano anche "neuromantico", a conferma dello scalpore suscitato da Gibson). I romanzi cyberpunk sono ambientati in un tetro mondo futuro popolato da personaggi sofferti e bizzarri, e caratterizzato da un impiego inquietante delle scoperte scientifiche. Buoni esempi di questo genere sono *Blade Runner*, *Max Headroom* e naturalmente *Neuromancer*.

La vicenda è ambientata nel 2058, nella tecnocittà giapponese di Chiba City. Per le strade pattugliate da inflessibili roboguardie d'acciaio non si vedono altri negozi se non quelli di hardware e software, inframmezzati di quando in quando da qualche lercio postribolo. I computer sono ormai talmente diffusi in ogni campo, che in molti luoghi pubblici esistono terminali per collegare le proprie attrezzature personali alle banche dati di tutto il mondo. La quantità d'informazioni disponibili ha raggiunto livelli tali che è stato necessario affiancare alle normali linee informatiche un nuovo sistema per tenere sotto controllo la mastodontica massa dei dati.

Questo sistema viene definito matrice primaria o, più gergalmente, cyberspazio: una vera e propria realtà alternativa, con la quale gli operatori possono interfacciarsi neuralmente, collegando direttamente il proprio cervello. Il cyberspazio prende così forma nelle loro menti come una griglia infinita, punteggiata da solidi geometrici (i vari database) che si scambiano continuamente informazioni rappresentate da segnali colorati che corrono in un cielo policromo.

Prima di riuscire a raggiungere questa incredibile e inquietantissima parte del gioco, tuttavia, dovremo passare un'infinità di guai e di avventure relative al nostro ruolo sociale di cowboy dichiarati. Questo ci porta ai chip tiche e difficoltà tecniche. Ma anche così, non è possibile accedere indiscriminatamente alle informazioni. Quelle che nel mondo reale vengono chiamate protezioni, come password o codici crittografici, nel cyberspazio appaiono come "ghiaccio" (in inglese "ice", che sta per Intrusion Countermeasure Electronics), una protezione che avvolge il solido geometrico respingendo attivamente gli accessi non autorizzati e colpendo fisicamente gli eventuali hacker. Al ghiaccio si sono poi unite negli ultimi tempi anche le Intelligenze Artificiali,

neurali: nel futuro tecnologico di *Neuromancer* è possibile acquistare a caro prezzo un particolare tipo di circuiti integrati che vengono installati direttamente nella nostra corteccia cerebrale, e che hanno il benefico effetto di fornirci istantaneamente diverse abilità particolari, non ultima quella di parlare e agire come un tutore dell'ordine.

Una volta ottenuto l'accesso al cyberspazio, è possibile muoversi su questi impulsi raggiungendo i più lontani database senza essere intralciati da lungaggini burocratiche

che vengono poste a difesa dei database più importanti (come ad esempio quelli militari o bancari). Le Intelligenze Artificiali possono sviluppare autonomamente i loro sistemi di difesa, evolvendosi e diventando in alcuni casi ben più intelligenti di un essere umano.

Tornando al mondo reale, si osserva subito che la maggior parte degli abitanti di Chiba City è dedita all'arte della penetrazione fraudolenta nelle banche dati. Questi hacker, che si autodefiniscono "cowboy", sfruttano le li-

nee tradizionali per un gran numero di attività illegali, tenendo nel frattempo una fitta rete di contatti per lo scambio di password, codici e programmi di sprotezione.

I pochi cowboy che riescono a permettersi un computer capace di penetrare nel cyberspazio (sono i più costosi fra i 40 che abbiamo a disposizione) sfruttano anche la matrice primaria per condurre i loro traffici, ma pochi giorni prima del nostro arrivo in scena sono accadute cose molto strane nel cyberspazio. I giornali elettronici sono pieni di notizie riguardanti cowboy assassinati da *qualcosa* che si nasconde sulla matrice primaria, e anche qualche utente regolare del cyberspazio è stato trovato col cervello carbonizzato, ancora collegato al suo cybercomputer.

In questa situazione, il giocatore impersona un hacker, deciso a scoprire cosa sta accadendo nel cyberspazio e possibilmente a renderlo di nuovo sicuro. L'impresa non è facile, dal momento che all'inizio del gioco ci troviamo senza computer, con pochi soldi in banca e debiti ovunque.

La situazione diventa presto meno drammatica quando, una volta recuperato il nostro computer dato in pegno, scopriamo una gran quantità di amici sui molti BBS della rete standard. Con il loro aiuto, e qualche piccola truffa, la nostra dotazione di software pirata e di password aumenta rapidamente, sino a che non scopriamo che qualcuno ha intenzione di ucciderci. Dopo molti arresti da parte delle roboguardie e un paio di visite sulle colonie extra-mondo nella nebulosa di Zion (e dopo l'incontro con il boss mafioso Julius Deane e la setta degli adoratori del videogame *Pong*), il nostro personaggio inizia a darsi da fare per procurarsi un cybercomputer.

I prezzi stratosferici lo obbligheranno certamente a vendere qualche organo o arto all'apposita Banca del Corpo, ma le sorprese che lo attendono nel cyberspazio valgono il sacrificio.

Neuromancer riesce a ricreare in modo eccellente l'atmosfera del romanzo, con le sue schermate animate e una notevole ricchezza di particolari, e questa è una qualità importante che dev'essere senz'altro sottolineata. Ma anche per quanto riguarda il gioco, contenuto su quattro facciate di disco, non possiamo che manifestare tutta la nostra approvazione.

La grafica, anche se in alcune animazioni si dimostra un po' lenta, è sempre nitida e spettacolare, un aspetto fondamentale in un graphic novel che si rispetti. E infatti la giocabilità del programma ne risente molto positivamente. Il sonoro non merita più di un cenno: è limitato a una canzone dei Devo che si ripete di quando in quando, e preferiamo riservare i nostri applausi al sistema di comando.

Il controllo del gioco è affidato a otto icone che, mediante alcuni sotto-menu, riescono a gestire ogni aspetto dell'azione senza problemi. Una volta penetrati nei database, la scelta più ovvia è quella di ricorrere alla tastiera, ma in ogni situazione è sempre possibile impartire i comandi via joystick.

Neuromancer è in questo momento il graphic novel più complesso fra quelli presenti sul mercato. La sua mappa di ridotte dimensioni lascia spazio a un evolutissimo sistema di database e a un cyberspazio che sembra tratto dalle migliori sequenze del film *Tron*. A differenza degli altri graphic novel, *Neuromancer* è rivolto soprattutto a giocatori interessati alla strategia e agli aspetti più tecnologici dell'avventura.

Un'ultima parola va spesa per segnalare le note umoristiche presenti in molte situazioni di gioco, dai dialoghi surreali che s'intrecciano con alcuni personaggi, ai numerosi riferimenti ad altri giochi. Su un database, per esempio, è possibile trovare *Battle Chess 2.0*, e una sequenza di viaggio spaziale è un'esilarante parodia di una scena pressoché identica contenuta in *Zak McKracken and the Alien Mindbenders*.

F.R.

DEATHLORD

Computer: C-64/128
Supporto: Disco
Prezzo: L. 22.000
Produzione: Electronic Arts
Distribuzione: C.T.O. (Via Piemonte 7/F, 40069 Zola Predosa – 051/753133)

Dalla Electronic Arts, ecco arrivare l'ennesimo gioco di ruolo: nella sua continua ricerca di produrre qualcosa di originale, pur rimanendo nel filone di maggior successo degli ultimi tempi, la casa americana ha pensato bene di dare a questo nuovo RPG un'ambientazione del tutto eccezionale.

Deathlord è indubbiamente un gioco di ruolo fantasy, ma c'è un particolare che lo rende unico: la trama non affonda le radici nelle solite leggende medievali europee, ma nella mitologia nipponica. Il mondo di *Deathlord* è infatti il Giappone feudale, benché gli autori lo abbiano ben mimetizzato sotto il nome di "Lorn".

Lorn è un arcipelago composto da 16 isole, governate dall'imperatore dell'isola di Kodan, Nakamoto. Nel momento in cui il gioco inizia, l'impero si trova in serie difficoltà a causa della presenza di un potente mago rinnegato che, dopo aver stipulato un misterioso patto con i Demoni, è tornato a Lorn con il titolo di Signore della morte.

Con questo pittoresco e terrificante nome, il negromante guida un gigantesco esercito di creature maligne alla volta di Lorn per prendere possesso dell'arcipelago. Le mostruose truppe del Signore della morte hanno già compiuto un raid distruttivo contro la maggiore città di Kodan dopo la capitale, scatenando il panico anche nella famiglia imperiale.

Vedendo il suo potere sempre più in pericolo, l'imperatore Nakamoto diffonde un editto rivolto a tutti gli avventurieri residenti nei suoi vasti domini perché trovino il modo di fermare il Signore della morte, promettendo a ogni componente del gruppo che tornerà vittorioso l'incredibile somma di 100 mila monete d'oro e il governo di un'isola. Come in ogni *quest* che si rispetti la missione potrà essere portata a compimento solo recuperando un certo numero di oggetti e d'incantesimi, oltre che naturalmente trovando il nascondiglio del misterioso e potentissimo Signore della morte.

Il gruppo di avventurieri controllato dal giocatore può arrivare a includere sino a sei componenti, scelti fra personaggi con diverse caratteristiche e abilità: oltre ai tradizionali attributi come Forza, Resistenza o Saggezza i personaggi di *Deathlord* mettono in mostra anche alcune capacità inedite. I membri del gruppo vengono scelti fra nove razze diverse, che determinano anche le rispettive aree di abilità.

La prima cosa che colpisce nella procedura di creazione dei personaggi (ovvero nella prima fase della partita) è la terminologia adottata. Come ogni termine rilevante del gioco, le razze di *Deathlord* mantengono i loro nomi giapponesi, disorientando un po' i giocatori abituati a trattare con elfi e nani. Se alcune razze come i Troll e gli Orchi hanno una corrispondenza automatica nella mitologia occidentale, termini come "Kobito", "Toshi" e "Nintoshi" riescono a spiazzare chiunque, se il giocatore non dedica almeno un po' di tempo allo studio delle regole del gioco. Una volta superato il disorientamento iniziale, però, ci si rende conto

che il programma offre una quantità di scelte ben maggiore di quelle rese disponibili dagli altri giochi di ruolo per computer, facilitando la creazione di un mondo particolarmente variegato e, in un certo senso, realistico.

Le specializzazioni di ogni personaggio sono addirittura 16, e vanno da ruoli abbastanza noti, come quello di Ninja o di Samurai a mestieri bizzarri come quello di Yabanjin (che corrisponde grossomodo al ranger dei giochi occidentali). Per non parlare degli incontenibili Kichigai. Ancora una volta, l'amplissima gamma di scelte permette la costruzione di personaggi perfettamente rispondenti alle proprie intenzioni.

La tavola relativa ad armi e armature comprende altri 18 vocaboli giapponesi, sotto i quali si celano tutte quelle armi spettacolari rese famose dai film di kungfu, come gli shuriken e le spade corte wakizashi.

Le ultime impronunciabili stranezze del gioco sono le 84 parole magiche disponibili per le quattro classi di personaggi che ne possono usufruire: in questo caso la situazione è addirittura allarmante, perché è obbligatorio scrivere il nome dell'incantesimo al momento di utilizzarlo. Senza bisogno di strani meccanismi, la necessità di avere sempre a portata di mano il manuale che le elenca è una semplicissima ed efficace misura anti-pirateria.

Una perfetta padronanza della magia è fondamentale per la riuscita del gioco, e gli utilissimi incantesimi (per usare gli alberi o l'erba come potenti alleati, oppure per teletrasportarsi in locazioni particolari e così via) fanno capire ben presto che i maghi sono elementi indispensabili nel gruppo.

Il mondo di Lorn è costellato di città, ma anche di cittadine, paesi e torri isolate: gli appassionati di giochi di ruolo più tradizionalisti saranno felici di sapere che su molte isole sono poi presenti tombe, templi... e gli immancabili dungeon. In ogni luogo si trovano poi personaggi con cui è possibile interagire, e ancora una volta *Deathlord* si discosta un po' dagli altri giochi di ruolo.

Nel dialogo con i personaggi che s'incontrano nel corso dell'avventura va prima di tutto impostato il tono (arrogante, umile, amichevole...), dopodiché si possono intavolare discussioni – proprio come accadeva nella serie *Ultima* – inserendo singole parole che rappresentano gli argomenti di conversazione. Inutile dire che il tono adottato provocherà risposte e reazioni diverse a seconda dell'interlocutore.

L'aspetto grafico del gioco ricorda molto da vicino proprio la serie *Ultima*, e in particolar modo il suo terzo episodio: gli ambienti e i personaggi vengono rappresentati con una schematica visione aerea composta di blocchi grafici monocromatici, che pur non entusiasmando particolarmente rende l'azione chiara e comprensibilissima.

Questa visuale non cambia nemmeno durante gli immancabili combattimenti: i 128 tipi diversi di avversari sono tutti rappresentati da una decina di semplici icone, e gli scontri vengono ridotti a messaggi di poche righe accompagnati da qualche flebile effetto sonoro.

Il sistema di comando di *Deathlord* prevede la tradizionale ridefinizione dei tasti per attivare particolari funzioni. Il programma consente tuttavia un'interessante novità: l'uso di macroistruzioni assegnate ai tasti funzione, che in pratica permettono al giocatore di ridefinirli a piacere. Questa opportunità si rivela in grado di semplificare notevolmente tutte le azioni che si presentano più volte nel corso di una partita, come per esempio l'acquisto di cibo o una particolare forma di attacco magico.

Per fare un esempio pratico, potremmo definire il tasto "F7" del nostro computer in modo da assegnargli la funzione "Acquista". In questo caso, premendo il tasto citato riempiremo il buffer dei comandi con la sequenza "seleziona un giocatore", "numero uno", "raccogli tutto l'oro", "parla", "a nord", "gentilmente". Com'è facile notare avremo così ridotto sei comandi a un'unica e semplice istruzione, che potremo ripetere a piacimento senza mai rischiare errori che, come ben sanno gli appassionati di role game, possono avere effetti catastrofici.

Queste "macroistruzioni" possono anche contenere parametri variabili, che rendono le funzioni assegnate ai tasti estremamente flessibili e paragonabili, fatte le debite proporzioni, a un vero e proprio lavoro di programmazione.

Indubbiamente, *Deathlord* ha molte particolarità positive oltre che singolari, ma purtroppo il gioco risente delle carenze in campo grafico e sonoro, che riportano alla mente il ricordo di una versione muta di *Ultima III*. Da segnalare inoltre l'uso massiccio e insistente della lingua giapponese anche in casi in cui non sarebbe affatto necessario.

Se da una parte questo contribuisce a creare un'atmosfera particolarmente originale e suggestiva, chi è abituato a giocare con prodotti più tradizionali ovviamente si troverà piuttosto confuso per le prime partite, e con ogni probabilità i risultati saranno catastrofici.

In definitiva, *Deathlord* rappresenta un prodotto decisamente atipico nella vasta famiglia dei giochi di ruolo, e per evitare inutili delusioni è bene essere avvertiti in anticipo: è un programma che può dare grandi soddisfazioni, ma solo dopo qualche tempo.

F.R.

DRILLER

Computer: C-64/128/Amiga
Versione: Amiga
Supporto: Cassetta/Disco
Prezzo: L. 25.000/29.000/59.000
Produzione: Incentive
Distribuzione: Lago (Via Napoleona 16, 22100 Como – 031/300174)

Quando la tecnologia "Freescape" fu applicata per la prima volta al C-64, recensori e giocatori gridarono al miracolo: per la prima volta nella storia dei videogame era possibile muoversi realisticamente in un mondo tridimensionale, completo di ombre e solidi "pieni". Il gioco che utilizzava questa notevole innovazione si chiamava *Driller*, e per parecchio tempo fu considerato in assoluto il miglior mondo digitale in cui avventurarsi, superiore anche alle notevoli simulazioni di volo della Sublogic per l'Amiga e altri famosi scenari computerizzati.

La tecnica Freescape si è recentemente evoluta con *Total Eclipse* (recensito in questo stesso numero), ma mentre i programmatori specializzati nel maneggiare i vari VIC II e 6510 del piccolo C-64 si scervellavano per aumentare del 15 per cento la velocità di aggiornamento dello schermo, gli specialisti dell'Amiga non sono stati certo con le mani in mano.

Si sa, la CPU a 16 bit dell'Amiga è ben più veloce dell'umile processore a 8 bit del C-64, e così la conversione dell'ormai mitico *Driller* sul computerone Commodore è risultata un vero capolavoro.

La trama è di quelle che coinvolgono e appassionano sin dal primo momento di gioco: il pianeta Evath vive sull'esportazione dei minerali estratti dalle sue lune gemelle, Mitral e Tricuspid. Mentre la seconda rappresenta una fonte ancora generosa di minerale, Mitral è stata abbandonata perché geologicamente instabile. Alcune imprese che operano ai limiti della legge hanno continuato a scavarla in lungo e in largo nonostante il rischio di far esplodere le sacche di gas che giacciono a pochi chilometri di profondità.

Questo stato di cose è perdurato sino a che gli osservatori astronomici di Evath non hanno rilevato la presenza di una cometa in rapido avvicinamento a Mitral. In condizioni normali, una semplice collisione non avrebbe causato che un piccolo cratere, ma Mitral è ormai priva di un'atmosfera che possa rallentare la cometa, e questo causa gravi preoccupazioni.

Il timore degli scienziati Evathiani è che l'urto causi l'esplosione di una sacca di gas e, tramite una reazione a catena, della luna stessa. Una catastrofe di questa portata altererebbe l'equilibrio gravitazionale dell'intero sistema planetario, provocando tremendi cataclismi anche sullo stesso pianeta madre.

Mentre le imprese minerarie illegali su Mitral abbandonano rapidamente il campo, le industrie di Evath lavorano febbrilmente alla realizzazione di un qualche marchingegno che possa mutare le sorti del pianeta. I progetti falliscono uno dopo l'altro, sino a che, a poche ore (reali!) dall'impatto con la cometa, non viene realizzato un veicolo telecomandato che potrebbe risolvere la situazione.

È a questo punto che il giocatore viene chiamato in causa: deve guidare il veicolo sulla superficie di Mitral, cercando di creare bocche di uscita per il gas che, una volta disperso nello spazio, non rappresenterà più un pericolo. Il veicolo che pilotiamo è una sorta d'incrocio fra un carro armato e un hovercraft, che scivola sul terreno girandosi e inclinandosi in ogni direzione, ma senza potersi per nulla sollevare da terra.

La luna di Evath ha una forma rozzamente geodetica, ed è divisa in 16 settori ben distinti. Nella confezione troviamo un'interessante mappa tridimensionale del pianetino, purtroppo quasi completamente in bianco.

Armati di molto coraggio e di un laser, iniziamo la nostra avventura. Se qualcuno si sta chiedendo a che cosa può mai servire un laser su di una luna disabitata, diremo che gli uomini delle società minerarie hanno "dimenticato", nella fretta della fuga, di disinserire i sistemi di difesa auto-

matici delle loro stazioni. Questi sistemi, che sono sostanzialmente torrette laser attivate dal calore (e quindi dalla vicinanza di qualunque mezzo a motore), rappresentano l'ostacolo più notevole che s'incontra su Mitral: bastano pochi colpi di laser per penetrare la leggera corazza del veicolo su cui ci troviamo. Fortunatamente, i nostri laser sono in grado di distruggere o almeno di cambiare la direzione di fuoco di queste torrette, e uno studio accorto della loro disposizione, unito a una certa abilità manuale, dovrebbe eliminare questa fonte di guai.

Gli altri problemi che s'incontrano sono essenzialmente di ordine geografico: il passaggio da un settore all'altro è infatti spesso subordinato all'attivazione di particolari meccanismi disposti nei luoghi più disparati, che si presentano sotto forma di veri e propri puzzle tridimensionali. Risolverli non è semplice, ma il successo sarà premiato con carburante extra o persino con la possibilità di trasformare il nostro veicolo in una vera e propria astronave.

Resta da dire delle trivellazioni, che pongono ancora qualche problema (come se non ne avessimo elencati abbastanza!). La perforazione del terreno di Mitral avviene teletrasportando una torre di trivellazione nel luogo in cui si trova il nostro mezzo, e richiede l'impiego di una grande quantità dell'energia a nostra disposizione. Dal momento che non tutti i punti di un settore sono adatti come possibili sfoghi per il gas, non è consigliabile fare i propri tentativi a caso: bisogna prima scoprire il punto esatto in cui piazzare la torre. Nel primo settore è molto facile (c'è addirittura una colossale "X" nel punto da perforare!), ma via via che ci s'inoltra nei settori più remoti di Mitral, il punto di trivellazione dev'essere individuato ricorrendo a strane incisioni o a mappe, oppure risolvendo complicatissimi puzzle.

Una volta che la torre è stata piazzata, l'estrazione del gas è automatica: pochi istanti dopo il teletrasporto, il computer di bordo ci comunica la percentuale di gas fuoriuscita, e se questa è superiore al 50 per cento il settore viene considerato sicuro.

La versione di *Driller* per l'Amiga è notevolmente superiore alla sua prima incarnazione per il C-64: la maggior velocità di aggiornamento schermo, le capacità grafiche e il sonoro del mostro a 16 bit hanno trasformato quello che era un gioco tipicamente strategico in un prodotto ai confini dell'arcade, dove tutto avviene con una velocità impressionante. Insieme alla velocità, è inoltre cambiato il sistema di comando, che non si basa più su strane sequenze di tasti come accadeva nella precedente versione. Il nuovo sistema, gestito da un pannello di controllo costituito da icone, rende la guida del veicolo decisamente semplice e intuitiva, e non capita praticamente mai di suicidarsi per errore come accadeva sin troppo spesso nella versione a 8 bit del gioco.

Per il sonoro, che già era impressionante ed estremamente suggestivo con il C-64, sono state sfruttate al meglio le capacità della nuova macchina: il risultato è una musica profonda e inquietante corredata da alcuni buoni effetti stereo, come nel caso degli spari delle torrette.

La tecnica Freescape impressiona come sempre, anche se titoli come *Starglider II* hanno raggiunto risultati ancor più spettacolari: la sensazione di presenza è incredibile, e poter girare attorno agli oggetti e osservarli da ogni prospettiva è affascinante. Nulla di strano se per qualche minuto si dimentica la propria missione mentre si esplorano nuovi angoli di Mitral. Una piacevole novità è data poi dalla possibilità di selezionare l'italiano come lingua per i messaggi di sistema, rendendo ancor più semplice l'approccio con il programma.

Driller è un programma che non dovrebbe mancare a nessun appassionato di strategia o di azione arcade, e probabilmente ripeterà il successo ottenuto nella versione a 8 bit... almeno sino all'arrivo della versione per l'Amiga del secondo titolo Freescape: *Dark Side*.

F.R.

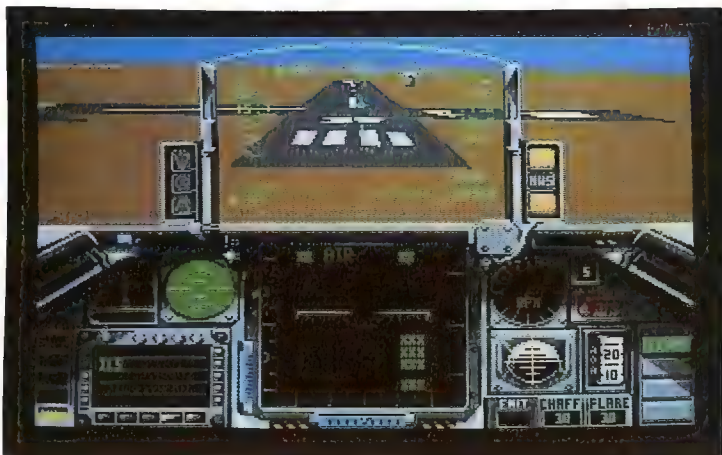
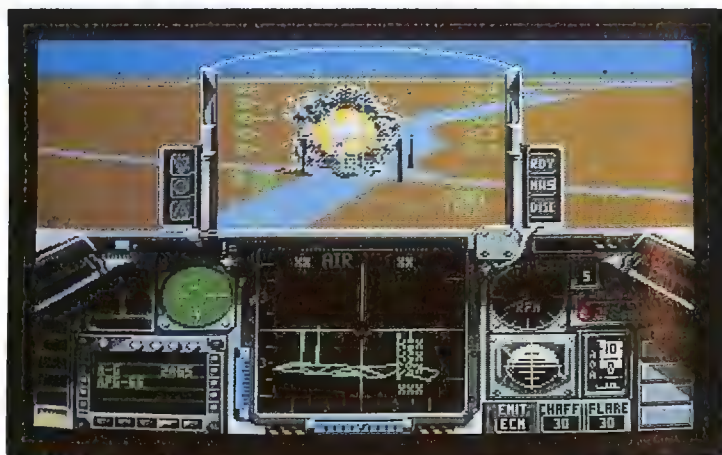
FALCON

Computer: Amiga
Supporto: Disco
Prezzo: L. 59.000
Produzione: Spectrum Holobyte
Disponibile presso: Lago (Via Napoleona 16, 22100 Como – 031/300174)

Le ali di un falco per il vostro computer? Sì oggi finalmente sono una realtà. Tutti gli aspiranti piloti da caccia, che non vedono l'ora di sedersi davanti al loro Amiga e volare, hanno la possibilità d'inserire nel loro hangar un nuovo straordinario velivolo, l'F-16 Falcon.

Dopo gli eccellenti risultati ottenuti dalla EA con *Interceptor* nessuno si aspettava che un pro-

dotto ancora migliore potesse essere realizzato così in fretta. E invece la Spectrum Holobyte ha creato *Falcon*, un simulatore di volo le cui qualità migliori sono l'elevata cura dei particolari e l'ottima giocabilità. Non appena ci si siede davanti al pannello comandi si ha un'intensa sensazione di realismo: non abbiamo esitazioni ad affermare che si tratta della migliore simulazione che si sia mai vista su una macchina Commodore, sia per la completezza della strumentazione, sia per l'ottima grafica. E dinanzi a un prodotto di qualità così elevata chi non eietterebbe senza paracadute fuori dal suo aereo Bruce Artwick (SubLogic) e il suo pessimo *Jet*?

Lo scenario nel quale si svolge

la simulazione è un'area divisa in nove quadranti, tutti occupati dal nemico tranne i due dove si trovano i bersagli per le esercitazioni, un lago, una catena montuosa e l'aeroporto del giocatore. Perlustrando il territorio nemico, oltre che montagne, un fiume, strade e aree agricole, troviamo due aeroporti, postazioni missilistiche, ponti ed edifici: ottimi bersagli per i nostri missili aria/terra e le nostre bombe. Tutto è stato realizzato con una buona grafica caratterizzata da superfici piene tridimensionali, e una soddisfacente cura per tutti i particolari.

Lo scenario è l'elemento fondamentale delle missioni con obiettivi al suolo. Prima di mettere le mani sui comandi, dopo aver scelto il nome del pilota, si accede a una schermata dalla quale bisogna selezionare una missione operativa. Sono possibili 12 scelte diverse. Per quel che riguarda le missioni con obiettivi al suolo si va dall'esercitazione di bombardamento con bersagli alla distruzione di ponti, strade, postazioni missilistiche, aeroporti ed edifici. Le altre missioni riguardano l'abbattimento di Mig-21 nemici. La difficoltà della simulazione va aumentando a seconda del grado del pilota (da tenente a colonnello). I parametri coinvolti nel grado di difficoltà sono tali e tanti – manovrabilità, accuratezza delle armi, possibilità di stallo, possibilità di perdita dei sensi del pilota, danni ai sistemi di bordo in caso di manovre errate... – che in realtà il numero delle missioni possibili va moltiplicato per i cinque possibili gradi militari.

Una seconda schermata permette la scelta dell'armamento. Oltre al cannoncino di bordo, sono disponibili due tipi di missili aria/aria, un missile aria/terra e due tipi di bombe. Rispettando i limiti di carico e tenendo conto delle disponibilità di magazzino, si possono anche aggiungere serbatoi di carburante esterni e un dispositivo per le contromisure elettroniche (per ingannare i radar nemici).

E ora, finalmente, un salto sul seggiolino dinanzi ai comandi. Come abbiamo già sottolineato, il pannello comandi è accurato e molto completo, e la sua struttura richiama da vicino il suo omologo reale. Gli indicatori riguardano pilota automatico, flap, aerofreni, freni idraulici, angolo di attacco, stallo, sensibilità dei comandi, accensione del reattore, carrello retrattile, contromisure elettroniche, incendio, lancio di missili nemici, ILS (Instrument Landing System), armamenti e postbruciatore. Non mancano orizzonte artificiale, contagiri e radar. Quest'ultimo fornisce anche tutta una serie di dati sui bersagli, che siano aerei o obiettivi al suolo. Lo HUD (Head-Up Display, un display proiettato dinanzi alla normale visuale del pilota) consente l'istantanea individuazione del bersaglio mediante dispositivi elettronici di puntamento. Lo HUD permette anche di rilevare numerosi dati relativi al Falcon, tra cui direzione, velocità e altitudine.

La rassegna sulla strumentazione potrebbe sembrare conclusa, a questo punto, ma non è così. Il nostro F-16 riserva numerose altre piacevoli sorprese. La semplice pressione di due tasti rivela infatti le parti laterali del pannello comandi, che risulta così ancora più esteso. Si tratta di un'ottima idea alla quale non possiamo che assegnare la lode. La strumentazione collocata sulla sinistra comprende trim, bussola, indicatori del carburante, velocità e corretta discesa dei carrelli. Sulla destra sono invece collocati ben 14 indicatori di avaria. Per quel che riguarda la visuale del pilota, è possibile guardare all'esterno nelle quattro direzioni. È disponibile anche la visuale dall'esterno (satellite, torre e altro aereo) che si rivela molto efficace e ben realizzata.

Il controllo del jet avviene tramite tastiera (ma sono consentiti anche joystick o mouse) e la risposta ai comandi è docile e fluida. Anche se la strumentazione del velivolo è piuttosto completa, non risulta per nulla difficile da imparare. Tutti i tasti sono infatti facilmente associabili alle funzioni che svolgono.

Le sensazioni che si provano ai comandi di questo apparecchio computerizzato sono estremamente realistiche. Tutto contribuisce a immergere il giocatore in un'atmosfera aeronautica, dalla

selezione degli armamenti – con un sergente dalla battuta sempre pronta – al motivo del *Silenzio* suonato in caso di morte del pilota. La noia, il nemico più temibile, è stata ampiamente combattuta con la varietà delle missioni e l'ulteriore differenziazione a seconda del grado del pilota. Si può iniziare con il grado di tenente e portare a termine tutti gli incarichi operativi, guadagnando nel contempo nastrini e medaglie. Se poi ci si stufa dell'eccessiva manovrabilità del jet si può passare al grado di colonnello, con il quale basta una manovra errata per perdere il controllo dell'aereo. Si può anche giocare in due collegando il proprio computer a quello di un amico (anche via modem) e dare vita a duelli aerei sempre più avvincenti.

Per abbattare i Mig nemici bisogna però conoscere le manovre elusive, alle quali è dedicato un particolare menu che permette di selezionare la manovra che si vuole imparare e... Ma abbiamo già detto troppo. Non vorremmo privare gli appassionati del piacere di scoprire personalmente le caratteristiche di questo meraviglioso aereo. Va ancora sottolineata l'estrema cura nella realizzazione, che siamo certi riuscirà a soddisfare persino i più esigenti aspiranti piloti militari.

La software house che ha realizzato il programma ha avuto un'idea davvero intelligente: una schermata in cui compare un breve testo che invita tutti coloro che posseggono una copia abusiva del programma *e lo apprezzano molto* ad acquistare l'originale. Questo per avere accesso al manuale (che è completo e ben fatto) e per permettere alla software house di realizzare in futuro nuove simulazioni sempre migliori. Chi scrive non può che associarsi a questo invito. Ne vale veramente la pena. Parola di pilota.

M.L.

TIMES OF LORE

Computer: C-64/128
Supporto: Cassetta/Disco
Prezzo: L. 25.000/29.000
Produzione: Origin (Microprose)
Distribuzione: Leader (Via Mazzini 15, 25020 Casciago – 0332/212255)

Uno degli ostacoli maggiori alla realizzazione di giochi di ruolo di qualità è sempre stata, e continua a essere, l'impressionante mole di dati necessari per costruire un mondo credibile. Spesso la qualità della grafica scade per favorire l'interazione con i personaggi o, viceversa, la varietà della storia è penalizzata se si cerca la "simulazione totale" di questo o quell'altro elemento di combattimento. A volte il desiderio di avere *tutto* ha generato veri e propri aborti digitali, sparsi su un numero incredibile di dischi. A poche settimane dalla comparsa dell'ignobile *Ultima V*, la presentazione di questo *Times of Lore*, sempre dalla Origin, ha suscitato non pochi dubbi ma... Come in tutte le storie fantasy, anche qui c'è un importantissimo "ma": le consolidate abitudini dell'industria del software hanno ceduto il passo a una svolta imprevista. Il buon Richard Garriott (presidente della Origin Systems) ha infatti stipulato un accordo con alcuni autori europei che, sfruttando le avanzatissime tecnologie di compattazione sviluppate durante la realizzazione dei numerosi giochi arcade prodotti in Inghilterra, hanno rivoluzionato in questo programma il concetto di role game computerizzato.

La più importante innovazione, messa in evidenza in ogni possibile spazio sulla confezione, sta nel formato del codice. *Times of Lore* è il primo gioco di ruolo veramente completo interamente contenuto in memoria! Immaginate le urla di gioia dei possessori di unità a nastro, e pensate a tutti i vantaggi che questo comporta anche per chi abitualmente usa il disk drive: scompaiono le attese, minime ma pur sempre fastidiose, che nelle serie *Ultima* e *Bard's Tale* interrompevano il gioco ogni manciata di secondi, e soprattutto si chiude l'era dei dischi da cambiare a ogni mossa!

I cosiddetti "esperti" staranno già arricciando il naso, convinti di trovarsi di fronte a un programma grezzo, brutto da vedere e da giocare, e dobbiamo confessare che questo è quello che abbiamo pensato anche noi prima di caricare il gioco. Ebbene, ricredetevi. *Times of Lore* non è soltanto bello da vedere (come potete facilmente constatare dalle foto), ma anche incredibilmente ben realizzato.

Andiamo con ordine. Prima del caricamento del gioco vero e proprio il giocatore ha due opzioni, dedicate l'una alla scelta del tipo di personaggio e l'altra a una breve introduzione allo scenario.

Selezionando la seconda opzione iniziano le sorprese: le schermate che compongono questa sezione reggono il confronto con quelle di certi programmi per l'Amiga (tenuto conto delle possibilità del C-64, ovviamente) e le musiche di sottofondo, composte dal bravissimo Martin Galway, sono fra le più belle sentite negli ultimi tempi. Un'analisi attenta dell'introduzione, nella quale vengono descritte le catastrofiche condizioni del regno senza re in cui ci troviamo, ci dà modo di scegliere il personaggio più adatto fra i tre che vengono presentati quando selezioniamo la prima opzione.

Lo scopo del gioco è di ritrovare Re Valwyn, scomparso da vent'anni insieme al figlio appena nato. L'impresa condurrà il giocatore a numerose missioni al servizio dei potenti del regno, nel corso delle quali potrà interagire con i 64 personaggi del gioco. Come nella miglior tradizione Origin, anche in questo programma è possibile dialogare con i molti abitanti del regno, questa volta utilizzando un intelligente sistema di menu variabili gestiti da joystick. La conversazione, anche se chiaramente pilotata verso una precisa direzione, risulta ancor più evoluta di quella di *Ultima V*, che sembrava aver segnato l'apice delle possibilità dei giochi di ruolo per computer in questo campo.

Se l'interazione con i personaggi è tanto buona allora sarà la parte grafica a farne le spese, dirà qualcuno. Ma anche in questo caso c'è da rimanere a bocca aperta: non solo la definizione degli elementi di gioco è tale da arrivare a distinguere ogni singola foglia su un albero, ma l'animazione dei 2.300 sprite del mondo digitale è addirittura incredibile!

Il fuoco scoppietta nei camini, i padroni delle taverne vagano da un tavolo all'altro lucidandolo e ritirando i piatti vuoti, le onde bagnano la spiaggia, gli uccelli volano in cielo, e potremmo continuare a lungo. Per trovare qualcosa di simile dovremmo citare i titoli più evoluti disponibili per il sistema Nintendo: dovunque si guardi c'è qualche sorpresa grafica, e sul C-64 la definizione è almeno cinque volte superiore...

Il parallelo con la console giapponese non è casuale, dal momento che, come spiega la stessa Origin nei suoi esaurienti comunicati stampa, *Times of Lore* è stato sviluppato appositamente per essere convertito per le console Sega e Nintendo, sia nei suoi elementi tecnici sia in quelli "letterari". La trama, studiata per conquistare l'utenza nipponica (che costituisce la parte più rilevante del mercato dei giochi di ruolo e dei giochi fantasy), risulta piacevole anche per i giocatori occidentali, spesso con un background televisivo costituito in gran parte da "manga", i cartoon giapponesi.

Tornando al nostro *Times of Lore*, un piccolo difetto è l'impossibilità di controllare più di un personaggio. È un'osservazione marginale, forse un po' pignola, ma resta il fatto che da sempre i giochi di ruolo si sono basati sulla collaborazione di numerosi personaggi con caratteristiche diverse, e queste vicende epiche perdono un po' del loro fascino quando vengono affrontate in solitario.

Times of Lore è comunque un programma grandioso, innovativo e tecnicamente ineccepibile, a cominciare dal caricamento, di ben 15 volte più rapido del normale. E subito dopo...

F.R.

CAPTAIN BLOOD

Computer: Amiga
Supporto: Disco
Prezzo: L. 40.000
Produzione: Infogrames
Distribuzione: C.T.O. (Via Piemonte 7/F, 40069 Zola Predosa - 051/753133)

Nello spiritoso racconto breve che troviamo con le istruzioni del gioco (fortunatamente tradotte in italiano), è evidente che i progettisti di *Captain Blood* hanno avuto non pochi problemi nel trovare un'ambientazione plausibile per questo bizzarro esempio di software. La storia ruota infatti intorno a uno scapestrato personaggio, Bob Morlok, un programmatore di videogame che si trova suo malgrado "risucchiato" dal suo stesso programma, e diventa una specie di navigatore solitario degli spazi digitali.

Il nostro Bob, dopo aver assunto il nome di Captain Blood, scopre di trovarsi all'interno di un'astronave bioelettronica gestita da un sarcastico computer di bordo: gli immancabili problemi presenti in qualsiasi gioco, questa volta sono dati dal fatto che Captain Blood non può separarsi dai sistemi di sostentamento vitale della nave, che tengono in vita con difficoltà il suo corpo bionico.

Blood sente che sta diventando sempre più debole, e dopo una breve indagine scopre che il passaggio nella dimensione digitale lo ha moltiplicato in un notevole numero di cloni, ognuno dei quali possiede parte della sua energia vitale. Per non rimanere completamente senza energie, Captain Blood deve quindi cercare i cloni sparsi per gli oltre 10 mila pianeti della galassia digitale in cui si trova ed eliminarli, riassorbendo così la propria energia.

Non potendo scendere di persona a esplorare i vari pianeti, Blood si affida al suo computer di bordo per trovare il modo di lanciare sonde intelligenti dalla nave sui vari corpi celesti, e la folle macchina trova una soluzione a dir poco originale. In men che non si dica, l'astronave di Captain Blood viene fusa bionicamente con un Oorx di passaggio, che sarebbe poi una specie di balena spaziale non troppo sveglia.

Il povero animale viene quindi fecondato artificialmente in modo da partorire continuamente piccoli Oorx, che potranno poi essere lanciati come siluri-sonda sulla superficie dei pianeti. Nello spazio digitale non si va troppo per il sottile, e ogni Oorx viene dotato anche di una carica esplosiva in grado di distruggere completamente i pianeti che dovessero rivelarsi pericolosi.

Quando il gioco comincia, Captain Blood ha già eliminato quasi tutti i cloni, e ne restano solo cinque nascosti in chissà quale remoto angolo della galassia. Essendo impossibile per ragioni di tempo esplorare metodicamente tutti i pianeti (Bob ha solo 45 ore reali per portare a termine la sua missione), è necessario trovare un sistema di ricerca più efficiente.

Questo sistema è senz'altro la caratteristica più originale di *Captain Blood*: un'interfaccia grafica che permette di dialogare mediante alcune icone con gli alieni presenti sui vari pianeti. Per prima cosa ci si deve portare nell'orbita di un pianeta, facendo uso dell'ampia mappa galattica che ci porta a destinazione semplicemente spostando due assi cartesiani sul punto desiderato. Un leggero tocco su uno dei pulsanti "organici" della console di comando, e ci troviamo proiettati in un coloratissimo iperspazio ipnotico e psichedelico che sembra tratto dalla vecchia serie di telefilm *Spazio: 1999*.

Arrivati nell'orbita del pianeta prescelto, dobbiamo dare un'occhiata alla stiva della nostra astronave per assicurarci che ci siano ancora Oorx disponibili. La stiva, come del resto tutta la nave, sembra essere tratta dalle scenografie "organiche" disegnate da Giger per *Alien*, e persino la sequenza in cui il piccolo Oorx viene partorito nella stiva è abbastanza realistica.

Il lancio dell'Oorx avviene senza troppi problemi, e prima di farlo scendere nell'atmosfera è possibile – tramite un apposito pulsante – farlo orbitare perché ci trasmetta una mappa del pianeta sottostante con due ingrandimenti alternativi. A questa funzione, particolarmente utile per identificare eventuali pianeti occupati da militari o difesi da laser, si può fare ricorso anche per rintracciare particolari strutture.

La discesa nell'atmosfera rappresenta l'unica sequenza arcade di tutto il gioco, ed è rappresentata con una strana grafica a metà strada fra quella vettoriale e quella frattale: sullo schermo appare la visione in soggettiva dell'Oorx, che dobbiamo dirigere con il mouse all'interno di un lungo canyon pieno di curve e stalagmiti. La manovra viene guidata da due frecce direzionali che appaiono in sovrimpressione sullo schermo, e una volta arrivati in fondo al canyon (se si è riusciti a evitare le difese del pianeta) le possibilità sono di trovarsi in uno spiazzo vuoto o al cospetto di un alieno.

Nel secondo caso, viene attivata automaticamente l'interfaccia simbolica di cui parlavamo prima: la parte inferiore dello schermo si riempie di piccole icone quadrate, delle quali viene resa disponibile solo una certa parte, a seconda dell'intelligenza dell'alieno incontrato. Il mouse ci permette di far scorrere la nutrita schiera di simboli a destra e a sinistra, e agendo sul puntatore otteniamo in una finestra la traduzione.

Il dialogo avviene selezionando una o più icone e premendo un eloquentissimo paio di labbra, che passeranno il nostro messaggio all'alieno. Le risposte della creatura arriveranno anch'esse sotto forma di simboli, e sarà così possibile gestire un dialogo anche piuttosto complesso con qualunque indigeno.

Una nota particolarmente positiva va fatta riguardo al lavoro svolto dalla Infogrames per adattare il programma ai vari mercati nazionali. Nonostante la versione originale del gioco fosse stata scritta per il mercato francese, infatti, *Captain Blood* viene venduto in ogni Paese con una diversa "interfaccia di comunicazione". Anche se l'icona con l'omino che si indica il petto ha un significato intuitivo, questo verrà tradotto come "Ich" in Germania, "I" in Inghilterra e "Io" nel nostro Paese. Certo avremmo preferito che questa politica di traduzione fosse stata adottata per altri prodotti di più difficile comprensione, ma vedere che anche le software house straniere iniziano a considerare l'Italia un mercato nel quale vale la pena d'investire non può che fare piacere.

Spesso, gli alieni ci chiedono di compiere per loro una missione in cambio delle informazioni richieste; una gran parte del gioco, in effetti, consiste nel portare a buon fine queste missioni, che in genere richiedono la distruzione di gruppi planetari.

Tra un lancio di Oorx termonucleari e un dialogo blandamente filosofico con bizzarri alieni, dovremmo riuscire a trovare le coordinate dei nostri cloni. Una volta trovati i falsi Captain Blood, sarà sufficiente utilizzare l'ultima opzione, relativa al teletrasporto a bordo di forme di vita senzienti e... consenzienti (infatti il soggetto dev'essere convinto a salire a bordo spontaneamente), per intrappolare il clone in una biocamera e recuperare così tutta la nostra energia.

Captain Blood è indiscutibilmente un gioco strano: strano per la trama, la grafica e la meccanica dell'azione. Tutti i suoi elementi sono molto ben realizzati, eppure la loro strana forma finale ha un sapore "alieno" che rende i rapporti con questo gioco molto più difficili del solito.

La grafica e la musica (di Jean-Michel Jarre!) sono di notevole qualità, e alcune sequenze – come per esempio la distruzione di un pianeta o il salto iperspaziale – sono veramente spettacolari. Questi aspetti, uniti all'idea del dialogo simbolico, più unica che rara, fanno di *Captain Blood* un programma molto interessante... a patto di non lasciarsi spaventare dal suo aspetto alieno! **F.R**

ELITE

Computer: Amiga
Supporto: Disco
Prezzo: L. 45.000
Produzione: Firebird
Disponibile presso: Lago (Via Napoleona 16, 22100 Como – 031/300174)

Nel 1984 la Acorn stava pensando di ridurre la propria produzione di computer a 8 bit, a causa dello scarso successo che stava ottenendo in patria ma soprattutto all'estero: pochi giorni prima che la decisione diventasse operativa, venne pubblicato proprio per l'Acorn BBC un gioco tanto innovativo, divertente e coinvolgente che trascinò sull'onda del suo successo lo stesso computer, le cui vendite alla fine aumentarono notevolmente.

Il titolo di quel videogame era *Elite*: si trattava essenzialmente di un simulatore di astronave, ma con qualche caratteristica che gli fruttò il titolo di "Gioco dell'anno", conferito dall'allora autorevolissima rivista inglese "Your Computer".

Dalla sua pubblicazione per l'Acorn BBC, *Elite* ha subìto molte conversioni, rimanendo tuttavia sempre fedele allo schema di gioco originario. Anche sull'Amiga, nonostante la grafica si sia trasformata da vettoriale pura a solida ombreggiata, la struttura del programma è sostanzialmente rimasta immutata.

Elite ci mette alla guida di una navicella commerciale armata di laser leggeri e dotata di una piccola stiva di carico. Il nostro compito è di sfruttare al meglio le capacità della nave in campo commerciale e militare, giungendo così alla qualifica di "Elite". L'azione di gioco è vista in soggettiva dall'interno della navicella, e il primo impegno a cui si trova di fronte il giocatore è quello d'imparare a governare l'astronave.

Con un buon uso dei comandi, affollati d'indicatori digitali a barre e da un radar tridimensionale, è possibile volteggiare nel buio spazio stellare e compiere qualsiasi genere di acrobazie. A differenza di molti altri giochi, *Elite* dà al giocatore la possibilità di controllare il proprio mezzo in uno spazio tridimensionale coerente, e per riuscire nel gioco è necessario imparare a gestire le rotazioni della nave su tutti gli assi.

Oltre ai comandi di volo tradizionali, la nostra nave ha la capacità di compiere salti iperspaziali (all'interno di una galassia) o intergalattici (da una galassia all'altra) sfruttando apposite carte per i rifornimenti, e con 18 diverse merci di scambio è possibile organizzare giri d'affari notevoli. Come nel mondo reale, il concetto che sta alla base delle transazioni di *Elite* dice che il prezzo di una merce aumenta in maniera inversamente proporzionale alla sua diffusione in un dato luogo. Per semplificare le cose, possiamo dire che un pianeta arretrato tecnologicamente pagherà grosse cifre per un computer, mentre un centro industriale sborserà molti crediti per prodotti agricoli e di allevamento.

Oltre alle merci citate negli esempi, i piloti più abili e coraggiosi possono provare a darsi al contrabbando, arte estremamente redditizia soprattutto se si commercia merce illegale come schiavi, armi da guerra o droghe. Esiste ovviamente la possibilità di vedersi confiscare il carico o di finire addirittura nella lista dei criminali ricercati dalla polizia.

Dal momento che per commerciare è necessario muoversi continuamente, gran parte del gioco è rappresentata dalla navigazione stellare, una futuristica forma di guida che presenta – chi l'avreb-

stellari. In ognuna delle otto galassie visitabili ci sono circa 250 pianeti, tutti visitabili, per un totale di 2 mila possibilità.

Una volta arrivata nell'orbita di un pianeta, la nostra astronave può agganciarsi a una delle basi astroportuali che si muovono come satelliti attorno a ciascun mondo. Gli astroporti vengono usati sia per rifornire la nostra nave sia per comprare e vendere le merci che ciascun pianeta offre e richiede. In *Elite* il commercio rappresenta il modo più semplice per guadagnare i crediti necessari

be mai detto? – pericoli e imprevisti. Tra l'altro è piuttosto facile incontrare durante i propri viaggi altri veicoli non molto ben disposti nei nostri riguardi. Spostandoci qua e là nello spazio possiamo imbatterci in una vasta serie di altre astronavi, ognuna dotata di caratteristiche particolari che vengono spiegate dettagliatamente nelle schede tecniche contenute nel nutrito manuale.

I piloti di questi veicoli sono generalmente umani ma fra loro possiamo incontrare anche una discreta varietà di creature aliene fra le quali i tremendi invasori

Thargoidi, che provengono addirittura da un'altra dimensione. Poi c'è la possibilità d'incappare in pirati interessati al nostro carico, bounty killer e altri bizzarri fuorilegge. Di qualunque genere siano questi incontri, i risultati sono sempre gli stessi: guerra!

Come in ogni gioco spaziale che si rispetti, anche in *Elite* i combattimenti avvengono a colpi di laser, ma investendo i propri guadagni in armi è possibile dotare la propria nave di qualche optional particolarmente utile nei conflitti più impegnativi. Nelle stazioni spaziali dei pianeti tecnologicamente più evoluti è possibile infatti acquistare laser di potenza maggiore di quelli standard, missili, sistemi elettronici di reazione automatica e persino una bomba a fusione in grado di polverizzare tutte le navi entro la portata del nostro radar.

Qualsiasi arma viene poi montata su uno dei lati della nave, da cui viene attivata selezionando la visuale opportuna Un laser da guerra posteriore, per esempio, è uno dei più convincenti "elementi di dissuasione" per eventuali inseguitori.

È evidente che potendo trasformare la nostra nave in un vero e proprio arsenale semovente, al giocatore si aprono anche carriere diverse da quella del semplice commerciante. Oltre a tendere agguati alle navi mercantili per poi raccogliere il carico dalle loro carcasse fumanti, chi gioca a *Elite* può per esempio passare dall'altra parte della barricata e arruolarsi nella polizia o, facendosi notare per le sue capacità, intraprendere una carriera di mercenario.

Quest'ultima possibilità viene trattata con particolare attenzione dal programma, che fornisce ai mercenari una decina di lunghissime "missioni speciali", conducendo la loro navicella nei quattro angoli dell'universo.

Una volta raggiunta una certa qualifica è infatti possibile ottenere missioni speciali di scorta, spionaggio e distruzione con le quali ci conquisteremo senz'altro l'ambito grado di "Elite".

Per merito della sua trama aperta a qualsiasi interpretazione da parte del giocatore, *Elite* è un gioco che affascina sin dalla prima partita, e dovrebbe risultare gradito sia agli amanti della strategia sia agli adoratori dell'ultraviolenza. La grafica piena della versione Amiga del gioco è veloce e discretamente realistica, nonostante il colore rosa dei "tremendi" invasori Thargoidi: durante il volo è facile calarsi nella parte del pilota alla Han Solo, grazie anche al realistico silenzio degli spazi infiniti.

L'unico commento sonoro offerto dal programma sono gli efficaci rumori di combattimento e i suoni del computer di volo... ma è possibile acquistare un computer d'atterraggio che si occupa delle manovre di aggancio con gli spazioporti al suono del *Danubio blu*, proprio come nella famosa scena del film *2001: Odissea nello spazio*.

Lo scenario di gioco è estremamente curato: ogni pianeta ha una propria particolare situazione politica e economica, e lo stile di vita della stazione spaziale ne viene influenzato. Per non parlare dell'aspetto delle diverse razze aliene e delle soluzioni logistiche bizzarre – dal punto di vista umano – che si scoprono di volta in volta nel corso del gioco.

Ma tutte le osservazioni che abbiamo fatto finora si potrebbero riassumere efficacemente in un'unica frase: un capolavoro del software di genere spaziale.

F.R.

THE MARS SAGA

Computer: C-64/128
Supporto: Disco
Prezzo: L. 35.000
Produzione: Electronic Arts
Disponibile presso: Lago (Via Napoleona 16, 22100 Como – 031/300174)

Dei tentativi della Electronic Arts di produrre giochi di ruolo originali abbiamo già fatto cenno nella recensione di *Deathlord*: purtroppo, si sa, anche nelle migliori famiglie si annidano le pecore nere, e questo *Mars Saga* è proprio uno di quei tetri ovini.

Lo scenario in cui è ambientato questo ennesimo RPG targato Electronic Arts è quel Marte tanto caro agli scrittori di fantascienza, su cui – dalle peripezie del John Carter di Edgar Rice Burroughs fino alle *Cronache marziane* di Ray Bradbury – si sono svolte centinaia e centinaia di avventure. Quest'ultima avventura elettronica vede il pianeta rosso colonizzato da poche migliaia di terrestri che conducono un'esistenza trasandata in grigie città sotterranee in cui la criminalità raggiunge livelli esasperati. Sulla superficie, altri esseri umani si aggirano fra le dune rosate: sono le tribù di nomadi composte da coloro che non riescono a vivere nei claustrofobici corridoi di plastica delle città, e vivono depredando gli sfortunati che passano nelle loro vicinanze. Oltre a questi predoni, Marte offre anche una varietà limitata ma estremamente pericolosa di fauna locale. Gli unici animali buoni, su Marte, sono quelli morti.

Il giocatore veste i panni di uno dei tanti reietti terrestri, emigrati per cercare facili guadagni e dimenticare le proprie sfortune passate. Ma evidentemente la sfortuna è dura a morire, perché avete perso la vostra navicella in un grave incidente nei pressi della città di Primus. Il gioco inizia all'interno del complesso sotterraneo di Primus; il vostro scopo iniziale è guadagnare qualche soldo per acquistare il minimo indispensabile a sopravvivere e magari anche una nuova navicella.

La possibilità di procurarvi il denaro necessario vi viene immediatamente fornita dal Supervisore della città: il vicino centro minerario di Proscenium non comunica ormai da diversi giorni con Primus, ed è necessario che qualcuno scopra che cos'è successo ai minatori. Diverse missioni ufficiali sono già andate perdute nel tentativo e, non essendo possibile sacrificare altri uomini qualificati, l'ultima speranza del Supervisore è riposta proprio in quei disperati che non hanno più nulla da perdere... cioè voi.

Come in ogni gioco di ruolo, è necessario reclutare altri personaggi che vi seguano nell'impresa. Un rapido giro dei bar di Primus sarà sufficiente per trovare un paio di rifiuti umani, che potranno poi essere tranquillamente scartati in favore degli uomini che in seguito vi si offriranno spontaneamente, affascinati dai vostri successi. Ogni personaggio è definito dalla consueta serie di attributi, indicati graficamente con alcune barre colorate, e da un certo numero di abilità particolari che si incrementano col procedere del gioco. In *Mars Saga* sono proprio queste abilità che contano: un personaggio con poche abilità speciali non andrà lontano, anche se possiede i migliori attributi del mondo.

Le abilità speciali, così come l'indispensabile denaro, vengono col tempo, in seguito ai continui combattimenti e alle piccole missioni secondarie ai limiti della legalità che vi verranno affidate da personaggi particolarmente potenti del sottobosco di Primus. Saranno proprio queste missioni che vi porteranno a visitare tutto il pianeta, facendovi scoprire a poco a poco il mistero che sta dietro alla "morte" di Proscenium.

Mars Saga è caratterizzato da un aspetto principalmente grafico e da un'interfaccia utente

gestita soltanto tramite joystick; ma anche se in genere queste caratteristiche sono considerate positive, qui rappresentano invece il punto debole del programma. Tutto il gioco è in realtà un grosso e confusionario labirinto, in cui il giocatore deve raggiungere un ipotetico punto A. Una volta arrivato in A, è obbligato ad andare in B, poi in C e così via, riducendo un intreccio potenzialmente interessante a un'insulsa ricerca che non dà alcuna soddisfazione. In pratica, tutto quello che dovete fare in *Mars Saga* è osservare l'effetto dei suoi ordini nella susseguente scena animata... ben poco entusiasmante, per quanto ben fatta.

Purtroppo, l'ansia della Electronic Arts di produrre il maggior numero possibile di giochi di ruolo per sfruttare al meglio il momento favorevole, ha portato alla creazione di un programma poco convincente e soprattutto poco divertente. Una saga della monotonia. Alcune soluzioni grafiche sono simpatiche (la finestra con la visione in soggettiva della nostra posizione, i combattimenti

esplorare mappe caotiche e approssimative (che il computer traccia diligentemente per voi) con la speranza di avere nel gruppo il personaggio in possesso dell'abilità speciale necessaria per accedere a una nuova parte di mappa, e continuando nel frattempo a combattere.

I combattimenti di *Mars Saga* hanno un forma prevalentemente grafica, nel senso che il giocatore – dopo aver impostato le azioni di ogni personaggio agendo tramite una sconclusionata serie di piccoli menu – non deve fare altro che con bombe a mano), ma in un gioco di ruolo "vero" la grafica è proprio l'ultimo elemento da considerare. Trama, definizione dei personaggi, interfaccia utente e così via, non sono che misere parodie di prodotti precedenti, e tutta l'originalità del gioco è stata affidata a brevi sequenze animate e a un paio di musichette decisamente poco "spaziali" che non aiutano certo *Mars Saga* a risollevarsi dalla mediocrità.

Peccato, Electronic Arts, da te una simile delusione non ce l'aspettavamo proprio. **F.R.**

TOTAL ECLIPSE

Computer: C-64/128
Supporto: Cassetta/Disco
Prezzo: L. 15.000/20.000
Produzione: Incentive Software
Disponibile presso: Lago (Via Napoleona 16, 22100 Como – 031/300174)

Da quando la tecnica di visualizzazione tridimensionale con superfici piene conosciuta come "Freescape" ha fatto la sua prima comparsa nel premiatissimo *Driller* (la recensione della sua versione per l'Amiga compare in questo stesso numero), i progettisti della Major Developments hanno migliorato notevolmente i suoi già eccellenti algoritmi grafici. Anche se i lavori di ottimizzazione non sono ancora conclusi, il Freescape è oggi più veloce del 15 per cento rispetto a quando venne usato per rappresentare la luna di Mitral, e i colori delle sue schermate non soffrono più di alcuna limitazione.

La Incentive, fiutata la possibilità di realizzare un nuovo supergioco, ha messo al lavoro i suoi programmatori, che sono riusciti effettivamente a produrre qualcosa di entusiasmante. *Total Eclipse* abbandona il filone spaziale di *Driller* e *Dark Side* e ambienta la sua storia proprio sulla vecchia Terra, in Egitto.

Pochi anni prima della caduta dei faraoni, come narra un'antica leggenda, un sacerdote di Ra, il dio del Sole, lanciò una tremenda maledizione sul popolo egiziano, che aveva perso la fede. Fece costruire un tempio consacrato a Ra sulla sommità di una piramide e ne ordinò la venerazione. Se qualcuno avesse osato distruggere quel simbolo della potenza di Ra, una tremenda catastrofe si sarebbe abbattuta sul mondo. Secondo la leggenda, il giorno in cui il sole non avesse più visto il tempio sulla piramide durante il suo passaggio, lo stesso Cielo

sarebbe precipitato sull'Egitto.

Il gioco inizia alle 8 del mattino del 26 ottobre 1930, soltanto due ore prima dell'eclissi totale che farà avverare la maledizione del sacerdote. Nei panni di un avventuriero alla Indiana Jones, il giocatore atterra con il suo biplano proprio accanto alla piramide, pronto a svolgere la sua missione: raggiungere la camera più alta del tempio e trovare il modo di placare la collera degli dèi.

Si entra nella piramide armati soltanto di una pistola; in dotazione al giocatore, inoltre, sono anche una borraccia e una bussola che si rivela preziosa per riuscire a orientarsi nel dedalo di camere all'interno della piramide. Come in tutti i giochi che sfruttano il sistema Freescape, l'azione è vista in soggettiva ed è comandata da tastiera. I 15 comandi a disposizione permettono di muoversi e guardare in ogni direzione, oltre che estrarre la pistola e usarla. Tre ulteriori comandi permettono di accucciarsi per passare in cunicoli particolarmente stretti e di riposarsi dopo azioni che hanno l'effetto di far accelerare il nostro ritmo cardiaco (indicato nel pannello informativo nella parte inferiore dello schermo).

Nel pittoresco interno della piramide il percorso non è dei più semplici. Oltre alla complicata disposizione di stanze e corridoi, il giocatore si trova anche di fronte a ostacoli di vario genere da affrontare in modo sempre diverso. Per esempio, le numerose porte sbarrate vanno aperte utilizzando come chiavi gli ankh (croci egizie che simboleggiano la vita), mentre bisogna tenere sempre sotto controllo la propria provvista d'acqua, reintegrabile attingendo da appositi bacili sparsi nelle camere della piramide.

Molte stanze contengono camminamenti pericolanti e trappole mortali, che vanno affrontate proprio come nei film di Indiana Jones: una delle situazioni più comuni è trovarsi di fronte a una statua che spara dardi avvelenati dagli occhi. La nostra pistola si rivela molto utile per rendere inoffensive alcune di queste trappole, come per attivare meccanismi altrimenti irraggiungibili.

Le camere della piramide non contengono solo pericoli, comunque. Spostando pannelli segreti ed esplorando attentamente ogni angolo della costruzione l'avventuroso esploratore sarà premiato con la scoperta di colossali tesori. Nel caso (piuttosto probabile) in cui non si dovesse riuscire a raggiungere il tempio, il numero di tesori raccolti nelle due ore reali di gioco sarà usato per valutare la nostra performance.

Total Eclipse rappresenta senz'altro una notevole evoluzione rispetto agli altri giochi Freescape: la mappa di gioco relativamente ridotta ha permesso una definizione maggiore degli elementi che la compongono, e la geometria delle stanze è plausibile e contribuisce a dare una forte impressione di realismo. Il 15 per cento di velocità in più rispetto a *Driller* non si fa notare molto, ma la migliore definizione cromatica è un notevole passo avanti per il Freescape.

Per quanto riguarda l'aspetto sonoro di *Total Eclipse*, va notato che la Incentive ha purtroppo abbandonato le affascinanti colonne sonore di *Driller* e *Dark Side* in favore di un fastidioso ritornello egizianeggiante, che non può essere eliminato se non abbassando il volume.

Un grosso difetto di *Total Eclipse*, che si riflette purtroppo pesantemente sulla sua giocabilità, è dato dal ritardo che intercorre fra comando ed esecuzione, che in molti casi si rivela fatale. La scarsità di riflessi della tastiera è tale da farci inciampare in fili che avevamo visto da tempo, o farci avviare implacabili verso baratri infiniti che non riusciamo a evitare nemmeno pigliando a martellate il tasto di "inversione a U". Come in *Driller*, tuttavia, una buona dose di pazienza è tutto quel che serve per imparare a domare il buffer, e una volta compiuta quest'impresa sarà possibile godersi con *Total Eclipse* uno dei più bei programmi d'esplorazione visti sul C-64.

F.R.

SIMULGOLF

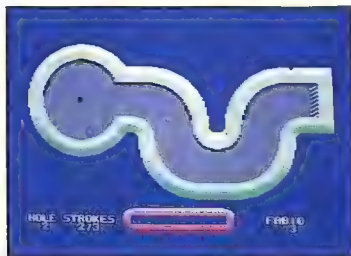
Computer: C-64/128/Amiga
Versione: C-64/128
Supporto: Cassetta/Disco
Prezzo: L. 22.500/35.000/39.000
Produzione: Simulmondo Italia
Distribuzione: Italvideo (Via A. Volta, 2/A – 40055 Castenaso – 051/784010)

"Simulmondo" è una nuova etichetta nel variopinto mondo del software. In questo periodo di proliferazione di nuove case, affiliazioni e joint venture, la notizia non sarebbe nemmeno degna di nota se non fosse per il fatto che la Simulmondo è una software house totalmente italiana. I prodotti Simulmondo, facilmente riconoscibili dal prefisso "Simul-" che compare sempre nel titolo, sono infatti progettati, scritti e distribuiti interamente in Italia. Questo, oltre che essere un grosso merito, in una nazione informaticamente arretrata come la nostra, si traduce in notevoli vantaggi pratici.

I giochi Simulmondo, infatti, presumibilmente arriveranno sugli scaffali dei negozi immediatamente dopo la recensione, eliminando il divario spesso abissale fra disponibilità teorica e pratica. Le istruzioni dovrebbero inoltre essere più comprensibili di certe ignobili traduzioni fatte negli stessi Paesi anglosassoni, e probabilmente (non abbiamo avuto occasione di provare) anche l'assistenza agli acquirenti sarà più affidabile di quella fornita da alcune case inglesi.

Viste le premesse, abbiamo caricato questo *Simulgolf* con una certa fiducia, rincuorati anche dal fatto che realizzare una versione digitale del minigolf non è certo un'impresa sovrumana, e lo sport in genere si presta bene a soluzioni simpatiche.

Mentre il computer caricava, è arrivata la prima delusione: il manualetto delle istruzioni era stilato in un linguaggio involuto e iniziatico, e parlava del programma come di un "interattivo" e dei suoi comandi come di "controlli dell'interattività". Tutta quest'interattività confonde un po', anche se fortunatamente il metodo di controllo del gioco è tanto semplice e intuitivo da non dare nessun problema, all'atto pratico. Meno male che non si trattava di un programma gestionale...

Una volta arrivati sul campo di gioco, vengono fatte le consuete scelte del livello di difficoltà e del numero dei giocatori (in inglese! I misteri delle software house sono insondabili) e l'azione comincia. Sullo schermo viene disegnato l'intero percorso da compiere per arrivare alla buca, e il primo giocatore piazza la sua pallina nell'area di partenza utilizzando un simpaticissimo braccio meccanico ronzante. Il joystick viene quindi usato per spostare un cursore nella direzione in cui si vuole lanciare la pallina, un'operazione che presenta qualche difficoltà quando si deve effettuare un tiro di precisione.

La forza del tiro viene calibrata premendo il pulsante e facendo così allungare una barra posta sul fondo dello schermo: rilasciando il pulsante la pallina viene lanciata con una forza proporzionale alla lunghezza della barra.

Il movimento della pallina, che spesso compie strane traiettorie piene di rimbalzi, è la seconda delusione: il calcolo dei rimbalzi "congela" brevemente l'azione ogni volta che la palla tocca un ostacolo, e l'effetto finale è molto poco realistico e decisamente irritante. Nel mondo simulato le palline interattive sembrano fatte di piombo, tra l'altro, come dimostrano le frenate improvvise del nostro proiettile. Strana la vita...

Di buca in buca e alternando l'azione fra i vari partecipanti, *Simulgolf* presenta poi i suoi pochi "ostacoli interattivi". A differenza del simpatico *Mini Putt* della Accolade, in cui gli ostacoli da superare erano numerosi, diversi l'uno dall'altro e spesso divertenti, qui le note originali si riducono al minimo. Abbiamo un razzo su binari che recapita la nostra pallina a destinazione, alcuni tunnel, qualche nastro trasportatore e uno strano sgorbietto che fagocita la pallina per sputarla poco lontano; le istruzioni interattive lo definiscono un "tulipano cannibale", ma noi rimaniamo dell'idea che si tratti di uno sgorbietto.

Il gioco prosegue monotono e silente (non c'è colonna sonora) sino a quando tutti i giocatori non hanno raggiunto l'ultima buca. A questo punto, una rapida occhiata al tabellone dei punteggi comunica il nome del vincitore, gli handicap e una serie di dati che farebbe la gioia di uno studente di statistica.

Simulgolf rappresenta un lodevole sforzo da parte della neonata Simulmondo, ma non riesce a reggere il confronto con i giochi di produzione anglosassone, nemmeno con quelli a basso prezzo. In tutte le sue parti traspare l'amatorialità del prodotto, che sarebbe stata accettabile soltanto se accompagnata da una maggiore varietà di gioco... e a un prezzo diverso. Gli amanti del minigolf possono sempre consolarsi con *Mini Putt*, mentre noi aspettiamo con interattiva pazienza nuovi titoli Simulmondo.

F.R.

SOFTWARE HELPLINE

Come affrontare il Dungeon Doom di Ultima V

Il dungeon si trova esattamente al centro del Continente Sotterraneo (Underworld), alle coordinate HA HA e si raggiunge scendendo dal Dungeon Shame. Usate l'amuleto per penetrare nell'area scura di Doom. Dovete avere con voi la corona, lo scettro e la Parola del Potere per Doom (Veramocor). Avrete anche bisogno dell'arco di sandalo (Sandalwood Bow).

Nel Dungeon Shame
Urlate (Yell) la Parola del Potere. Sulle prime vi sembrerà di essere bloccati, ma attaccando le rocce sulla parete sud con le asce magiche scoprirete un'uscita segreta. Uscite e imboccate il primo corridoio a sinistra. Seguitelo sino ad arrivare alla discesa al settimo livello. Un cartello vi informerà che siete arrivati al livello "più profondo". Ignoratelo, perché andando quattro volte verso sud scenderete all'ottavo livello. Una volta arrivati laggiù, muovetevi a ovest di una casella e lanciate l'incantesimo Des Por contro l'Underworld. L'uscita, se vi interessa, si trova (partendo dal vostro punto di arrivo nell'ottavo livello) due caselle a ovest, una a sud, una ancora verso ovest, due verso sud e infine tre passi in direzione est (attraverso una porta segreta). Per trovare Doom, viaggiate verso est più che potete, poi usate l'incantesimo "Blink" sino alle pianure erbose. Attraversate l'acqua sul Tappeto Volante (Magic Carpet).

Prima di entrare in Doom dovete avere ucciso tutti i Signori delle Ombre (Shadowlords). Dal momento che in questa zona non si possono usare gli incantesimi di Up e Down, l'unico modo per uscire è di liberare Lord British o di farsi uccidere.
L'Anello dell'Invisibilità (Ring Of Invisibility) qui non funziona molto bene, ma aiuta molto indossare la Corona (Crown).

Nel Dungeon Doom
Livello Uno
Uscite dalla stanza segnata con "a" camminando verso nord sino al limite della parete ovest e usando lo Scettro, che fa sparire i muri. Seguite i corridoi sino alla scala in (b).

Livello Due
Da (c) arrivate in (d) e scendete usando la scala.

Livello Tre
Da (e) muovetevi continuamente a est sino a oltrepassare una porta segreta e raggiungere un corridoio. Seguitelo, e potrete scendere in (f).

Livello Quattro
Scendendo dal Livello Tre apparite in (g), ma dovreste continuare a scendere in modo da raggiungere il Livello Cinque.

Livelli Cinque e Quattro
Vi ritrovate in (h): andate a est e risalite il pozzo in (i). Vi troverete sopra al centro della sala a forma di croce (j) del Livello Quattro. Andate a sud e frugate (Search) per trovare un passaggio in (k). Scendete nel pozzo, e vi ritroverete in un corridoio est/ovest nel Livello Cinque. Se usate una Gemma, vedrete il livello come una griglia infinita. In questo momento vi trovate fra le sue intersezioni.
A ogni intersezione c'è un pozzo che porta in alto. Andate a est sino al primo incrocio, girate a nord e prendete il primo pozzo che va in alto fra gli incroci (che non è indicato sulla mappa). Dovreste ritrovarvi in (m) al Livello Quattro. Frugate per trovare il pozzo in (l) nella zona dell'angolo di sud ovest, che vi porta a una scala che scende attraverso tutto il Livello Cinque vi fa arrivare in (n) sul Livello Sei.

Livello Sei
Andate verso ovest da (n) sino alla stanza all'incrocio, poi muovetevi a nord sino alla scala e scendete.

Livello Sette
Da (o) andate a sud nella stanza in mezzo al corridoio. Non attraversatela! Prima di uscire dovete eliminare tutti i mostri presenti, e in questo caso un incantesimo di Paura (Fear) seguito da una Scossa Tellurica (Tremor) o qualcosa di simile fa miracoli. Assicuratevi d'indossare la Corona. Dopo aver eliminato tutti i nemici, raccogliete le loro attrezzature. Nel mezzo della stanza, troverete nascosto sotto la sabbia un cofanetto. Apritelo, prendete tutto quello che contiene e sotto apparirà una scala. Scendete. (Se siete in condizioni critiche, prima di scendere lasciate la stanza e curatevi).

Livello Otto
Apparite in una stanza (p) divisa in due da un fiume. Uccidete i draghi e i serpenti di mare (l'incantesimo Fear funziona molto bene anche qui). Per attraversare il fiume, dovete attivare un meccanismo che si trova nell'angolo in alto a sinistra della stanza. Usate un'ascia magica, una lancia o un'accetta da lancio e colpite il muro in quella sezione. Il punto esatto è a sinistra delle due cascate. Un ponte si abbasserà sul fiume non appena farete centro. Un effetto secondario della manovra è di fare apparire alcuni mostri, ma se non avete voglia di affrontarli potete correre velocemente attraverso il ponte e liberarvene. Muovetevi verso ovest e scendete la scala in (q) per arrivare al Livello Sette.

Livello Sette
In (q) dovete andare a sud ma non c'è uscita in quella direzione. Per risolvere il problema, eliminate i mostri e spingete il muro vicino alla grata per attivare un meccanismo, che apre una parte di muro verso nord. Dal punto in cui si trova la grata, muovetevi verso la parte di muro appena modificata e spingete, e aprirete finalmente un passaggio verso sud. Dopo essere andati in quella direzione, cercate (Search) per trovare una trappola alla fine del corridoio. Il pozzo che troverete vi porta giù in (r), al Livello Otto.

Livello Otto
Muovetevi verso ovest da (r) e risalite la scala per trovarvi in (s) al Livello Sette.

Livello Sette
Da (s) andate a ovest e a sud, cercando il passaggio verso il basso nell'angolo. Muovetevi attorno, cercate nuovamente e infilatevi nel pozzo successivo (t) per scendere in (u) al Livello Otto. (Assicuratevi d'indossare la Corona prima di scendere).

Livello Otto
La prima cosa da fare sull'isola (u) è uccidere tutti i mostri. Scegliete un personaggio da mandare nella sezione settentrionale della stanza, che presenta un'uscita. Quando l'uscita è scomparsa, qualcuno dovrebbe spingere la piccola nicchia nelle pareti dell'angolo in alto a destra. Tutto ciò fa apparire un'ulteriore uscita verso est. Andate a est in (v) e

risalite la scala per raggiungere (w) al Livello Sette.

Livello Sette
In (w) c'è un gigantesco braciere. Non colpite o spingete il braciere (a meno che non vi diverta particolarmente l'idea di annegare nella lava). Per eliminare i demoni che si trovano dall'altra parte della barriera, usate lo Scettro per cancellarla. (Se indossate la corona, i demoni non possono farvi del male.) Ora attraversatela e distruggete i mostri. Un'alternativa è restare vicini al muro e usare una Morningstar, anche se questo vi farà perdere parecchio tempo. Uscite a est, ma cercate bene in giro prima di arrivare in fondo al corridoio per evitare di cadere in una trappola. Dopo aver cercato, potete scendere nel pozzo in (x) sino al Livello Otto.

Livello Otto: Finalmente liberi!
Questa stanza (x) non ha uscite. Camminate sino allo specchio... sempre che abbiate con voi l'Arco di Legno di Sandalo.

Pool Of Radiance

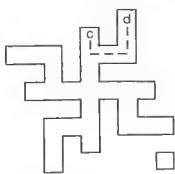
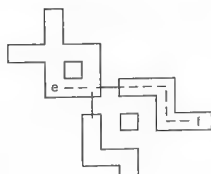
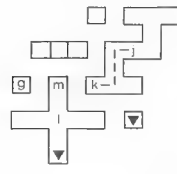
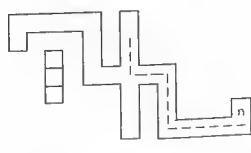
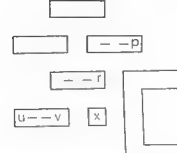
Per moltiplicare gli oggetti in vostro possesso (solo nella versione per il C-64), dovete innanzi tutto fare una copia del disco di salvataggio e usare questa copia.
1) Date tutti gli oggetti da duplicare a un unico personaggio. **2)** Andate alla Training Hall e usate l'opzione Remove Character su quel personaggio. **3)** Lasciate la Training Hall. **4)** Accampatevi e salvate la partita. **5)** Spegnete il computer. **6)** Ricaricate e richiamate il gioco appena salvato. **7)** Aggiungete al gruppo il personaggio appena eliminato. **8)** Partite con Begin Adventuring. **9)** Accampatevi nuovamente. **10)** Spostate gli oggetti dal personaggio aggiunto di recente agli altri membri della banda. **11)** Selezionate "Alter" e "Drop Character" eliminando il solito personaggio. **12)** Salvate il gioco. **13)** Tornate al punto 5 e ripetete il tutto tante volte quante volete duplicare gli oggetti di cui al punto 1. **14)** Dopo aver finito le moltiplicazioni, salvate il gioco con tutti i vostri personaggi.

Prima di ripulire il cimitero (Graveyards), assicuratevi di avere un sacco di parole di Restoration e la Bottiglia degli Efreeti (Efreeti Bottle) presa nelle Caverne dei Coboldi (Kobold Caves).

Nel Valjevo Castle, i maghi dovrebbero memorizzare Palla di Fuoco (Fireball) e Proiettile Fulminante (Lightning Bolt). Usate Palla di Fuoco sui giganti delle colline (Hill Giants) e Proiettile Fulminante sui giganti di fuoco (Fire Giants). Prima di andare dagli uomini lucertola (Lizard Men), assicuratevi di aver liberato gli schiavi sull'isola dello Stregone (Sorcerer's Isle). Sul pannello nello Studio di Yarash, il bottone blu vi teletrasporta in un punto vicino all'uscita, mentre gli altri tre vi portano in una serie di camere piene di tesori.

Finale del gioco: Nel Castello di Valjevo, ci sono quattro caselle nel centro del labirinto. Una contiene un falso Tyrantsaxus. Ripulitelo e scoprirete che una delle pareti è un'illusione. Passatevi attraverso e troverete delle scale che portano ad altre stanze, una delle quali contiene il vero Tyrantsaxus.

Ultima V

Ecco un sistema rapido per trovare pozioni, pergamene, armi, gemme ecc. Per cominciare trovate le Skull Keys. Osservate l'armaiolo (Armourer) a Minoc (Shenstone) a mezzogiorno. Verso l'una di pomeriggio, cercate (Search) l'albero al quale è sempre vicino e troverete cinque chiavi. (Queste vengono sostituite ogni giorno all'una, così potete tornare a prenderne quante ne volete). Scendete nelle fondamenta del castello di Lord British e usate una Skull Key per aprire la porta più a sud, dove troverete tre cofani. Prendete tutto quel che contengono, uscite dalla stanza e risalite una scala qualsiasi. Scendete e rientrate nella stanza, e i cofani saranno nuovamente pieni. Vendete tutto quello che non vi serve. Poiché rubare abbassa il vostro status di Avatar, entrate in un tempio relativo a un'avventura già risolta, meditate per tre cicli e offrite 900 monete d'oro. Per ogni 900 monete, viene alzato di uno il vostro livello di Avatar. ■

INPUT/OUTPUT SVELA I SEGRETI DEL VOSTRO COMPUTER

C-64/128 in modo 64

***190 Memoria disponibile in tempo reale -** Questo programma trae spunto da un'applicazione di pubblico dominio per l'Amiga e viene in aiuto ai programmatori dando una continua informazione su un importante parametro del Commodore 64: la memoria occupata da un programma Basic e dai suoi dati.

L'idea viene messa in pratica molto semplicemente, agganciando alla routine di interrupt una routine che calcola e visualizza periodicamente la quantità di memoria disponibile (dalla fine degli array numerici fino all'inizio delle variabili stringa, le quali, com'è arcinoto, vengono sistemate a partire dal fondo della memoria).

Come tutti i programmi che visualizzano dati sullo schermo in maniera asincrona rispetto al programma in esecuzione, anche questo comporta lo svantaggio di disturbare la prima riga dello schermo: una possibile soluzione potrebbe consistere nell'inserire le informazioni dentro uno sprite (si veda l'Input/Output n. 170 orologio-sprite pubblicato sul numero 4/88 di *Commodore Gazette*).

```
100 for i=49152 to 49222
110 read a: poke i,a
120 next i
130 sys 49152
140 data 120,169,13,141,20,3,169,192,141,21,3,88,96
150 data 165,162,41,31,208,24,162,4,56,165,51,229
160 data 49,8,32,46,192,40,165,52,229,50,32,46,192
170 data 169,36,141,0,4,76,49,234,72,41,15,32,57,192
180 data 104,74,74,74,74,24,105,48,201,58,144,2,233
190 data 57,157,0,4,202,96
```

Commodore 128

***191 Creazione di file sorgenti -** Ecco un metodo semplice quanto efficace per ottenere dei file di testo dai nostri programmi Basic o in linguaggio macchina. Com'è noto, questo tipo di programmi viene memorizzato sul disco in un formato caratteristico del Commodore 64/128 ossia come insieme di codici compattati (token) nel caso del Basic o come insieme d'istruzioni in linguaggio macchina nel secondo caso: questo ci impedisce di trasferire con facilità tali programmi su altri computer, oppure di inviarli attraverso le linee telefoniche, oppure ancora di analizzare o stampare con un word processor il listato del programma. Le brevi procedure presentate soddisfano questa esigenza.

Per programmi Basic:

```
open 2,8,1,"<filename,s,w>":cmd 2
list [riga iniziale]-[riga finale]
print# 2:close 2
```

Input/output rivela ogni mese ai suoi lettori routine di programmazione per C-64, C-128 e Amiga. Avete in mente un'idea particolare, una routine, un utile stratagemma di programmazione, avete sviluppato un breve programma o in generale pensate che le Vostre conoscenze possano interessare l'utenza Commodore? Allora scrivete, e inviate gli eventuali listati stampati su carta bianca o salvati su disco a:

COMMODORE GAZETTE
Input/output
Via Monte Napoleone, 9
20121 - Milano

Per programmi in LM:

```
open 2,8,1,"<filename>,s,w":cmd 2
monitor
d [indir. iniziale] [indir. finale]
x
print# 2:close 2
```

Si noti che l'utilità di questa conversione nel caso di programmi in linguaggio macchina è ancora maggiore, dal momento che i file sorgente risultanti possono venire modificati attraverso un word processor e quindi sottoposti nuovamente a un assembler (per esempio *geoAssembler*).

Commodore Amiga

***192 Un comando per stampare -** A chi proviene dal mondo MS-DOS spesso capita di provare a utilizzare il comando "print" sull'Amiga, ottenendo come unico risultato un messaggio di errore: in quell'attimo, dopo l'iniziale stupore, cui fa seguito un tentativo di capire perché il disco sistema non è nel disk drive richiesto, ci viene in mente che la questione degli standard è ancora tutta da risolvere e che quindi dovremo utilizzare qualche altro sistema per ottenere la nostra hard-copy.

In linea di massima, il metodo più usato per ottenere la stampa di un file consiste nel farne una copia verso il dispositivo PAR:. Vi proponiamo un file comandi che svolge tale operazione, e anche un trucco che vi consentirà di mandare in esecuzione dei file comandi senza utilizzare il comando EXECUTE.

File script "print":

```
.key filename/a
run COPY <filename> TO par:
```

File "pr.c":

```
#include <stdio.h>
#include <functions.h>
main(argc,argv)
int argc;
char *argv[];
{ static char comm[80]="EXECUTE print ";
  strcat(comm,argv[1]);
  Execute(comm,stdin,stdout);
}
```

Il trucco consiste nell'impiego della funzione C Execute(), che manda in esecuzione un comando CLI. Nel nostro caso, poiché ci interessa eseguire un file comandi, la stringa di comando passata alla funzione Execute() sarà costituita da "EXECUTE <nome file script> <parametro 1>".

Ovviamente si poteva ottenere la stampa di un file anche in modo più semplice, ma qui abbiamo voluto evidenziare una particolare funzione dell'AmigaDOS. ■

SoftMail

VENDITA PER CORRISPONDENZA DI PROGRAMMI ORIGINALI PER TUTTI I TIPI DI COMPUTERS
VIA NAPOLEONA 16 - 22100 COMO - TEL. (031) 30.01.74
® SoftMail è un marchio registrato da Lago snc

SoftMail è l'azienda di vendita per corrispondenza con il più vasto assortimento di programmi originali per tutti i computers.
L'organizzazione professionale e la serietà del nostro servizio garantiscono la massima cura per ogni ordine. Non solo: da quest'anno i nostri clienti più fedeli possono usufruire di offerte speciali e promozioni esclusive. Prova ad effettuare un acquisto tramite il tagliando qui sotto: ti sentirai subito un cliente molto speciale! SoftMail ti riserva informazioni, consigli, anticipazioni e tutta la gentilezza che ti meriti.

Presentiamo in questa pagina alcune tra le ultime novità del catalogo SoftMail.
Ecco qualche informazione utile sul nostro servizio: è possibile effettuare ordini telefonici dopo aver effettuato un primo ordine scritto. Se desideri notizie sulla disponibilità ed i prezzi dei prodotti che non compaiono in questa lista puoi telefonare allo (031) 30.01.74 dalle 14:30 alle 18:00 dal lunedì al venerdì.
SoftMail può organizzare la consegna anche tramite corriere: interpellaci per maggiori informazioni. Oltre alle ultime novità qui esposte, SoftMail offre l'intero catalogo delle seguenti case: Aegis, Cinemaware, EA, Firebird, Microprose, Rainbird, SSI, SSG, Sublogic.

ACCESSORI

Copritastiera A500	25.000
Final cartdrige III	110.000
Flicker master	29.000
Joy. SpeedKing A.F.	33.000
Joy. Tac 5	39.000
MouseMat tappetino	22.500
Coprimouse	20.000
Portamouse	12.500
Portadischi 3" (30)	34.000
Portadischi 5" (40)	37.000
SlimLine (tastiera64)	49.000

LIBRI/HINTS & TIPS

Alternate city clue specificare 8/16 bit	18.000
Bard's tale I	22.500
Bard's tale II	25.000
Bard's tale III	25.000
Dungeon master	25.000
Elite	18.000
Pool of radiance	20.000
Quest for clues	39.000
Ultima V	22.500
Wasteland clue	16.500

AMIGA

Animation multipl.	125.000
Barbarian II	telef.
Battlechess	49.000
Carrier command	49.000
Comic setter	139.000
Comic setter art disks	telef.
Def con 5	59.000
Deluxe paint/print comb.	tel.
Deluxe photolab	250.000
Deluxe print II	telef.
Director	99.000
Disk drive esterno	299.000
Dos2Dos	75.000
Dragon's lair (1 Mb)	75.000
Driller ITALIANO	59.000
Dungeon master	59.000
Elite	45.000
Empire	49.000
F16 Falcon	59.000
Fish	45.000
Flight simulator II	99.000
Scenery disks	telef.
Forms in flight II	199.000
International soccer	39.000
Italy 90 soccer	39.000
Jet & Japan bundle	110.000
Joan of arc	telef.
Lattice 5.0	telef.
Light!Cam!Action!	99.000
Mickey mouse	25.000
Modeler 3D	129.000
Murder on Atlantic	59.000
Neuromancer	telef.
Photon video:	
Cell animator	199.000
Transp.controller	490.000
Phantom fighter	45.000
Prowrite 2.0	179.000
Publisher plus	125.000
Purple saturn day	45.000
Reach for the stars	59.000
Rebel..Chickamauga	99.000
Rocket ranger	59.000
SEUCK	telef.
Sculpt/Animate 4D	825.000
Skychase	59.000
Starglider II	49.000
Sword of sodan	69.000
The bard's tale II	49.000
The works!	249.000
Triad Vol. 1	59.000
Tracker	49.000
TV sport football	59.000
Ultima IV	49.000
Univ.Military simul.	45.000
VIP professional	199.000
Virus infect. protect.	89.000
Who fr. Roger Rabbit	telef.
Willow	59.000
Zak McKracken	59.000
Zoetrope	199.000

COMMODORE 64/128 DISCO

Accolade tutti gli sports	tel.
Adv.dungeons&dragons	tel.
American civil war III	49.000
Barbarian II	25.000
Defender ... Crown	15.000
Driller	29.000
Echelon (c/ LipStick)	59.000
Exploding fist+	18.000
Fish	29.000
Home video produc.	69.000
Mars saga	35.000
McArthur's war	49.000
Microprose soccer	49.000
Neuromancer	39.000
R-type	25.000
SEUCK	35.000
Stealth mission	75.000
The bard's tale III	35.000
Times of lore	25.000
Total eclipse ITA	20.000
Ultima V	49.000
Wasteland	39.000
Who fr. Roger Rabbit	telef.
WWF superstars 1 e 2	tel.
Zak McCracken	39.000

ORA APERTI AL PUBBLICO

Buono d'ordine da inviare a: LAGO DIVISIONE SOFTMAIL, VIA NAPOLEONA 16, 22100 COMO, TEL. (031) 30.01.74, FAX (031) 30.02.14

Desidero ricevere i seguenti articoli:

Titolo del programma	Computer	Prezzo
G	Contributo spese di spedizione Lit.	5.000
ORDINE MINIMO LIT.20.000 (SPESE ESCLUSE)	TOTALE LIT.	

☐ Pagherò al postino in contrassegno
☐ Addebitate l'importo sulla mia ☐ CartaSi ☐ Mastercard ☐ Visa ☐ American Express
Numero ____________________ scad. ________

Cognome e nome ____________________
Indirizzo ____________________ Nr. ______
CAP _____ Città ____________________ Prov. ___ Tel. ________

FIRMA (Se minorenne quella di un genitore)
Verranno evasi SOLO gli ordini firmati ____________________

SISTEMI ESPERTI PER L'AMIGA

L'INTELLIGENZA ARTIFICIALE SULL'AMIGA E IL PACCHETTO MAGELLAN

Una carrellata sui sistemi esperti, le shell, i metodi di analisi, le reti neurali e i frame. Ora finalmente possiamo fare esperimenti sull'IA anche con l'Amiga, grazie a Magellan

di Matthew Leeds

Nel film *2001: Odissea nello spazio* compariva un sofisticato computer, HAL 9000, capace di comprendere ordini espressi a voce e di rispondere nello stesso modo. Veniva descritto come un sistema dotato d'intelligenza artificiale, che alla fine impazziva e uccideva quasi tutto l'equipaggio dell'astronave che gli era stata affidata. Se torniamo sulla Terra, scopriamo che la nascita dell'intelligenza artificiale è stata molto faticosa anche al di fuori delle sale cinematografiche.

La tecnologia dell'intelligenza artificiale può essere suddivisa in tre aree principali: la comprensione del linguaggio naturale, la robotica e i sistemi esperti. La comprensione del linguaggio naturale, insieme a quella del linguaggio parlato, ha come meta la realizzazione di una macchina simile a HAL 9000 (senza malfunzionamenti). La robotica si occupa di apparati meccanici controllati tramite computer. Con il nome di "sistemi esperti", infine, si indicano tutti i programmi che, grazie a una base di conoscenze, possono giungere a conclusioni simili a quella di un esperto umano.

Quanto "esperto" può essere un sistema esperto? L'American Express fa uso di un moderno sistema computerizzato in grado di prendere decisioni autonome in merito a pratiche di credito insolite. La base di conoscenze di questo sistema è stata creata consultando gli esperti dell'azienda. Per ora, gli errori sembrano essere minimi, anche se non vengono forniti dati ufficiali. La Digital Equipment si dimostra invece meno riservata, e dichiara risparmi annuali che vanno dai 70 ai 100 milioni di dollari grazie all'uso di programmi "intelligenti" nella realizzazione di sistemi informatici complessi. Il risparmio è dovuto principalmente a una riduzione di 300 volte nel tempo di progettazione.

In ogni caso, di sistemi esperti ce ne sono molti. Alcuni dimostrano capacità di decisione indubbiamente elevate, ma altri si limitano a mettere in mostra un comportamento intelligente in settori piuttosto ristretti. In effetti il termine "sistema esperto" può creare false illusioni. Proprio per questo, da un po' di tempo è entrata nell'uso un'espressione

La schermata di apertura del sistema esperto Magellan

più generica: “sistemi basati sulla conoscenza”. Si ritiene che questa definizione sia più adatta per fare riferimento a pacchetti software che utilizzano aspetti della conoscenza umana e del ragionamento per analizzare i problemi e trarre le loro conclusioni.

Hard coded e shell

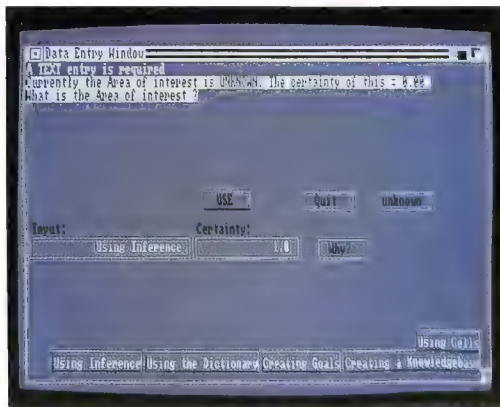
I primi sistemi di questo tipo sono stati i cosiddetti sistemi “hard coded”, ossia programmi destinati ad avere a che fare con un unico insieme d’informazioni. Fu soltanto con la creazione di *Mycin* (un sistema esperto progettato da Edward Shortliffe della Medical School di Stanford per aiutare i medici nella scelta degli antibiotici da somministrare ai pazienti) che si comprese l’importanza di mantenere separato l’insieme delle conoscenze dal meccanismo logico. Questo portò alla creazione di una “shell” per i sistemi esperti, ovvero un dispositivo software d’impiego universale in grado di utilizzare informazioni di ogni tipo.

Portare a termine *Mycin* richiese un impegno equivalente a 20 anni del lavoro di un uomo, e alla fine conteneva circa 500 regole su cui basare le sue decisioni. Diventò più preciso degli stessi medici che gli avevano fornito le regole su cui si basava, ma non riuscì mai ad avere il tono rassicurante di un vecchio medico di famiglia.

Le shell dei sistemi esperti, o motori inferenziali, si basano su regole costruite tramite il classico costrutto condizionale IF... THEN. La sezione relativa alla condizione “IF” si chiama antecedente, e la sezione relativa all’implicazione “THEN” si chiama conseguente. Possono esservi anche diversi antecedenti in una proposizione, e devono tutti risultare veri perché la conseguenza venga considerata vera. Il motore inferenziale memorizza ciascun antecedente immesso dall’utente in una raccolta di dati da considerare veri (asserzioni). A mano a mano che esamina le asserzioni a lui fornite, il motore inferenziale può applicare le regole per trarre alcune conclusioni da considerare a loro volta vere e quindi da aggiungere alla sua base di conoscenze. Alla fine giunge a una soluzione.

Passi fondamentali nella creazione di un sistema esperto

Un’importante differenza tra i sistemi hard coded e le shell dei sistemi esperti consiste nell’ordinamento delle regole: i programmi del primo tipo richiedono al programmatore la conoscenza anticipata di ogni possibile asserzione, nell’esatto ordine in cui intervengono. Nelle shell, che sono più flessibili, l’ordinamento dei dati è ininfluente perché è lo stesso motore inferenziale che provvede a mettere nella giusta successione i dati che gli servono. E lo fa nel momento stesso in cui crea la linea di ragionamento per rispondere a una domanda o risolvere un problema. In altre parole, è il meccanismo decisionale stesso a orientarsi da solo nel cammino da seguire attraverso le regole e le conoscenze acquisite.

Inoltre, non è necessario che ciascuna regola porti a una conclusione finale: una regola può semplicemente appartenere a un gruppo di condizioni che devono risultare vere perché si raggiunga una certa conclusione. Ognuna delle regole va analizzata seguendo un certo ordine per vedere se le sue condizioni sono soddisfatte dalle asserzioni di base (vere per

Nel modo Data Entry il sistema esperto interroga l’utente

definizione) e dalle conclusioni raggiunte fino a quel momento. Questo processo prosegue finché tutte le regole non sono state esaminate e nessuna fornisce più conclusioni vere, né si verificano mutamenti nelle conoscenze di base. Oppure finché non si trova una soluzione.

I due metodi di deduzione

Il motore inferenziale può esaminare le regole in due modi distinti, in avanti o all'indietro (forward chaining e backward chaining). Nell'analisi in avanti si parte da un insieme di dati noti e si seguono le regole secondo il loro percorso logico, dalle premesse alle conclusioni. Alla fine si ottengono una serie di conseguenze (metodo "data driven"). Usando l'analisi all'indietro, invece, il sistema parte da un'ipotesi di lavoro e segue il processo a ritroso, dalle conclusioni alle premesse, ottenendo infine dati analoghi a quelli contenuti nella sua base di conoscenze: questa viene considerata una prova della correttezza dell'ipotesi di partenza (metodo "object driven"). Un esempio di base conoscitiva può essere il seguente:

Regola 1 – Se B è vero, allora C è vero.
Regola 2 – Se C è vero, allora D è vero.
Regola 3 – Se A è vero, allora B è vero.

Risolviamo un problema con il metodo forward chaining.

Asserzione iniziale: **A è vero.**
- Secondo la regola 3, se A è vero, anche B è vero.
- Secondo la regola 1, se B è vero, anche C è vero.
- Secondo la regola 2, se C è vero, anche D è vero.

Dunque D è vero.

Risolviamo lo stesso problema con il metodo backward chaining.

Ipotesi: **D è vero.**
- Applichiamo la regola 2: D è vero se C è vero.
- Applichiamo la regola 1: C è vero se B è vero.
- Applichiamo la regola 3: B è vero se A è vero.
- Asserzione iniziale: A è vero.

Dunque D è vero.

Il metodo forward chaining parte dalle asserzioni riguardanti un problema e attraversa tutte le possibilità che scaturiscono dall'analisi dell'insieme di regole. Alla fine vengono tratte le conclusioni: non ci sono informazioni che possano stabilire a priori quali conclusioni si otterranno dal processo deduttivo, o se una particolare conclusione risulterà vera. Si tratta di un sistema utile soprattutto quando le conoscenze richieste per arrivare a una decisione sono tutte disponibili sin dall'inizio e quando le conclusioni attese sono in numero inferiore alle premesse.

Il metodo backward chaining sulla base di un'ipotesi esamina le regole all'indietro, dalla conclusione alle premesse

Magellan chiede conferma per associare un oggetto a un valore

Il metodo backward chaining, invece, fa uso delle risposte che si suppongono vere per un dato problema ed esamina all'indietro i dati in suo possesso per giungere alle asserzioni di base. Il percorso attraverso le regole è guidato dalle conclusioni, anziché dalle premesse, e la risposta viene considerata esatta se al termine dell'analisi il sistema trova asserzioni iniziali che coincidono con quelle fornite come vere.

Sebbene l'idea su cui si basa questo secondo metodo sia fondamentalmente semplice, la sua codificazione a livello software è molto complessa. Analizziamo il problema in dettaglio: l'analisi si conclude subito se il sistema trova

un'asserzione che coincide con la supposizione di partenza; in caso contrario il sistema deve andare alla ricerca di una regola la cui conseguenza contenga la supposizione da controllare. Se tutti gli antecedenti di quella regola sono veri, il processo termina, altrimenti si deve effettuare un'analisi ricorsiva sulle altre regole finché non si ottiene la conclusione attesa. Il metodo backward chaining può essere più efficiente di quello forward chaining perché se la supposizione iniziale è corretta, durante il processo analitico/deduttivo si spreca meno tempo nei vicoli ciechi. In pratica, risulta il metodo da preferire se l'utente fa una buona scelta riguardo alla possibile soluzione e quando ci sono più soluzioni di quante sono le asserzioni iniziali.

Nella shell di un buon sistema esperto, i due metodi vengono combinati insieme per ottenere una maggior flessibilità: l'analisi in avanti stabilisce che cosa si può scoprire analizzando le conoscenze a propria disposizione, mentre quella all'indietro determina la strada da seguire per trarre le conclusioni.

Un buon motore inferenziale deve inoltre fornire una spiegazione del percorso (ovvero del "ragionamento") seguito per raggiungere la conclusione: può trattarsi di un elenco ordinato delle regole di cui ha fatto uso, oppure di un database separato contenente le informazioni necessarie per capire che regola o che informazione sta alla base di ogni scelta. Questa spiegazione è fondamentale perché nessuna persona competente potrà mai accettare le conclusioni di un sistema esperto che non sappia giustificare il proprio cammino logico. Un sistema di spiegazione bene organizzato può anche essere utilizzato come strumento didattico.

Altri tipi di sistemi "intelligenti"

I sistemi sin qui esaminati, quelli basati sulle regole, sono un po' limitati, perché devono analizzare ripetutamente e nell'esatto ordine l'intera base di conoscenze. Di conseguenza, l'efficienza del processo deduttivo diminuisce con l'aumentare delle regole. Con ampie basi conoscitive il processo diventa tanto lento da non essere più adatto ad applicazioni che richiedono risposte immediate. Questa lentezza può essere ridotta o eliminata del tutto facendo uso di metodi di ricerca mirati, come l'organizzazione delle regole in una rete di raggruppamenti, nella quale l'utilizzazione di una certa regola limita la successiva analisi a un sottogruppo di regole. Oppure si possono ordinare le regole facendo sì che la regola appropriata a un dato contesto venga rintracciata per prima. Qualsiasi ordinamento, però, diminuisce la reciproca indipendenza dei dati della base conoscitiva.

Il sistema esperto è giunto a una conclusione e la espone

Magellan può visualizzare un'immagine IFF durante il processo deduttivo, o al momento di esporre la conclusione

Chiariamo il concetto con un semplice esempio. Al mattino dobbiamo decidere come vestirci. La nostra scelta deve soddisfare due esigenze: essere adeguata alle condizioni atmosferiche e al nostro umore. Anche se disponiamo di un guardaroba limitato, dobbiamo fare i conti con un notevole numero di combinazioni possibili tra i vestiti, e sprecheremmo moltissimo tempo se ogni mattina dovessimo prendere in considerazione tutte le possibilità e stabilire

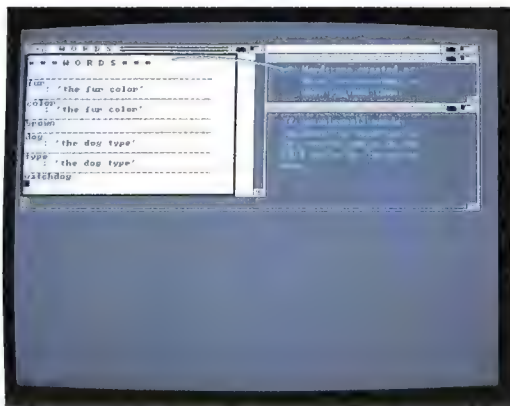
in che misura ognuna soddisfa le due esigenze. Però possiamo operare una drastica selezione sulle possibili scelte andando a vedere com'è il tempo, ed eliminando per esempio tutti gli indumenti pesanti se c'è il sole. In tal modo il campo di ricerca si riduce notevolmente, e può essere ridotto ancora eliminando subito le scelte che non si accordano per nulla con il nostro umore: le possibilità ancora in gioco dovrebbero essere ormai molto poche.

La quantità d'informazioni che può essere introdotta in una singola regola senza che diventi troppo complessa per un essere umano è limitata. D'altro canto, la suddivisione delle regole in un gran numero di sotto-regole più elementari porta inevitabilmente a un rallentamento del sistema deduttivo. Si ricordi inoltre che i sistemi esperti basati sulle regole, non possono analizzare i problemi da diversi punti di vista come farebbe un uomo. Devono per forza limitarsi soltanto al punto di vista per cui sono stati progettati.

Ci sono comunque altri approcci possibili per un sistema esperto, al di là di quello basato sui connnettivi logici IF... THEN. Quest'ultimo, infatti, benché sia ottimo per i sistemi esperti basati sull'esperienza, richiede una base conoscitiva troppo specifica. Per trattare dati generici serve un approccio diverso, se non si vuole che il numero di regole diventi infinito. La logica e i sillogismi, per esempio, sono una base su cui costruire questo diverso approccio. Il sillogismo è un costrutto logico che si presta splendidamente al meccanismo deduttivo. Eccone un esempio: tutti gli uomini sbagliano; Matthew Leeds è un uomo; dunque, Matthew Leeds sbaglia.

Un'altra possibile strada è quella delle reti neurali, in cui le informazioni sono concatenate da legami di parentela. Anziché regole, vengono creati elenchi di oggetti o di caratteristiche, e vengono definite le relazioni tra i vari elementi. All'aggiunta di nuovi elementi, il programma può "chiedere informazioni" e/o cercare d'identificare il nuovo elemento. A questo scopo viene usato un equivalente software della ricerca di ripetute conferme alle proprie ipotesi, analogamente a quanto avviene nel processo d'apprendimento umano.

Il sistema deve sempre saper giustificare le sue conclusioni

Di recente si è diffuso un altro metodo, conosciuto come "metodo dei frame". I frame sono "luoghi" in cui vengono inserite descrizioni generiche di oggetti, ciascun attributo dei quali occupa una casella. Un insieme di caselle definisce un concetto, un oggetto oppure un evento. I frame rappresentano la conoscenza: possono essere più efficaci delle strutture convenzionali quando vengono organizzati in forma di alberi o di reticoli, perché sfruttano il modello gerarchico per ereditare gli attributi da altri frame. Così, un frame che si trova alla radice dell'albero rappresenta un concetto o un oggetto al massimo livello di astrazione, mentre i frame sistemati sui nodi periferici (o foglie) rappresentano conoscenze specifiche (le esperienze del sistema). Maggiore è il livello a cui si trova un frame sull'albero, meno specifico è il suo tipo di conoscenza. Sistemi esperti di questo tipo sono in grado di operare meglio rispetto agli altri, soprattutto nel caso di una base d'informazioni piuttosto esigua.

Le word sono gli elementi utilizzabili per la base di conoscenze

Magellan: un sistema esperto per l'Amiga

I sistemi esperti, disponibili solo da pochi anni per i piccoli computer, in genere hanno un prezzo che oscilla dai 3 ai 5 mila dollari. Il pacchetto *Magellan* per l'Amiga viene venduto a un prezzo molto inferiore, soltanto 195 dollari. Offre un motore inferenziale strettamente basato sulle regole, che fa uso di un linguaggio abbastanza vicino all'inglese per la costruzione di ulteriori regole. Consente inoltre l'uso di fattori probabilistici per la rappresentazione dei vari livelli di certezza, sia per quanto riguarda i dati forniti dall'utente, sia per le risposte del sistema. È accluso anche un dettagliato sistema per la descrizione della linea di ragionamento seguita dal programma.

Magellan consente l'impiego di entrambi i metodi per l'analisi delle regole, forward e backward chaining, e la scelta è affidata all'utente. È scritto in linguaggio C, e la sua velocità di esecuzione è da tre a otto volte superiore a quella che si ottiene con analoghi sistemi basati sul linguaggio LISP. Sui computer Amiga da 512K, si possono definire da 50 a 100 regole, mentre su sistemi da un megabyte si possono superare le 500 regole. Per la scelta delle risposte standard alle richieste del sistema sono disponibili gli strumenti tipici dell'interfaccia utente dell'Amiga (gadget, menu...).

Il pacchetto *Magellan* comprende un manuale di 160 pagine e due dischi contenenti il programma e un database d'esempio. Dopo il boot del programma ci si ritrova all'interno della schermata principale suddivisa in due finestre: quella superiore è la finestra di controllo, mentre quella inferiore è riservata al dialogo con il programma. A questo punto si possono fare tre cose: creare una nuova base di conoscenze, modificarne una già presente in memoria oppure caricare nel computer una base di conoscenze già pronta.

Per creare una nuova base conoscitiva occorre definire un set di regole attraverso la finestra RuleBuild. Questa mette a disposizione un insieme di caselle (cell) nelle quali vengono inseriti gli elementi di ogni regola: Object (il soggetto della regola); Attribute smi del motore inferenziale una certa dose di incertezza.

La finestra RuleBuild permette all'utente d'inserire le regole

type (la categoria a cui appartiene il Value che assegneremo al soggetto); Operator (le relazioni tra Attribute e Value); Value (l'attributo assegnato al soggetto). Si possono poi stabilire delle "soglie" e dei livelli d'incertezza per ognuno di questi elementi. Infatti, dal momento che i problemi del mondo reale non hanno quasi mai soluzioni sicure al cento per cento, per simulare la realtà è necessario inserire nei meccanismi del motore inferenziale una certa dose di incertezza.

Ogni terna di Object, Attribute e Value forma una casella. Ciascuna casella può essere messa in relazione con variabili, file di testo, immagini in formato IFF, animazioni o parti di programma. In questo modo è possibile associare alla base conoscitiva una serie di messaggi, visivi o di diverso tipo, che il sistema utilizzerà per sottoporci le sue domande nell'analisi all'indietro, o per mostrarci le soluzioni nell'analisi in avanti.

Magellan può visualizzare testi e immagini, o può chiamare altri programmi per documentare le conclusioni a cui giunge

A questo punto è utile una precisazione: se questa descrizione sul funzionamento di *Magellan* appare un po' vaga, è soltanto

perché il processo stesso, sotto molti aspetti, è piuttosto vago. Se qualcuno vi chiedesse come fa un bambino a imparare, sareste in grado d'illustrare il processo con una serie di passi precisi? La creazione di una base di conoscenze avviene in modo molto simile. Si comincia con un argomento che si ritiene di conoscere bene, quindi si procede alla creazione di un elenco di regole che definiscono i rapporti interni tra i vari elementi che compongono il soggetto che ci interessa. A questo punto s'interroga il sistema per vedere se fornisce risposte corrette alle nostre domande e spesso capita di scoprire che nelle risposte ci sono errori, anomalie e strani equivoci logici che non ci saremmo mai aspettati. Dobbiamo quindi intervenire per modificare le regole.

Con un po' d'immaginazione, è possibile pensare alla creazione di una base di conoscenze come a un complesso videogame, nel quale bisogna comprendere a fondo un insieme di meccanismi, definirli in termini di costrutti logici di tipo condizionale (IF... THEN) senza dare mai niente per scontato. Ma la difficoltà maggiore nasce dal fatto che non dobbiamo batterci contro un avversario computerizzato, ma contro noi stessi. Non c'è niente di più arduo e di più frustrante che tentare di dare un ordine logico alla propria conoscenza e trasformarla in codici che una macchina possa utilizzare.

Il futuro di Magellan

Una nuova versione di *Magellan* dovrebbe già essere disponibile nel momento in cui questo articolo viene pubblicato: la versione 1.1 conterrà diverse migliorie, come il supporto dello standard video europeo PAL, nuovi moduli d'interfaccia, miglioramenti nella gestione dei requester per i file e per la stesura delle regole; conterrà inoltre il supporto per una porta *ARexx* e per la scheda con i processori 68020/68881. I moduli d'interfaccia aggiuntivi comprenderanno: SBFIO, per l'impiego dei dati in formato *Superbase*, WKSIO per la conversione dei dati creati dal *Lotus 1-2-3* in formato WKS, ASCIO, utile per gestire file di

Le caselle contengono un oggetto, un attributo e un valore

testo (per questo modulo sarà presente anche il codice sorgente in linguaggio C), SERIO, per controllare un modem e contattare i BBS, e PORTIO, un programma *ARexx* che consente al pacchetto *Magellan* di essere controllato da un altro programma.

Si possono creare gadget che contengono risposte standard

La Emerald Intelligence, la casa produttrice del programma *Magellan*, ha organizzato una gara, il First Annual Magellan Application Contest. La gara è suddivisa in tre categorie: la miglior applicazione orientata alla produzione, la migliore applicazione creativa e la più bizzarra.

Ci sarà un vincitore assoluto e due finalisti per ogni categoria: il primo premio è un monitor Moniterm da 19'' e un pacchetto *Magellan*, mentre ai finalisti saranno assegnati altrettanti pacchetti *Magellan*. La data di scadenza per le iscrizioni è il primo giugno 1989. ■

Per ulteriori informazioni contattare direttamente:

Emerald Intelligence,
Magellan Contest Rules
3915-A1, Research Park Drive
Ann Arbor, MI 48108 (USA)
(tel. 001/313/663-8757)

ABBONATI!

A CASA TUA UN REGALO OGNI MESE

ABBONARSI CONVIENE!

- Sicurezza di non perdere neanche un numero
- Prezzo bloccato per un anno
- Sconto del 15% sul prezzo di copertina

- Possibilità d'includere nell'abbonamento gli arretrati
- Comodità di ricevere la rivista direttamente a casa

Per chiunque sia interessato ai computer Commodore, *Commodore Gazette* è indispensabile. Nessuna rivista in Italia offre ai suoi lettori tanta qualità con recensioni hardware e software, listati, presentazioni esclusive, informazioni di ogni genere riguardanti C-64, C-128 e Amiga.

Un abbonamento a *Commodore Gazette* è il regalo più bello che possiate fare a voi stessi e agli altri... un regalo nuovo ogni mese. Ma non è finita! Risparmierete il 15% sul prezzo di copertina e potrete includere nel prezzo dell'abbonamento anche i numeri arretrati che mancano alla vostra raccolta. Resta inteso che per ogni arretrato scelto verrà spedito un numero di prossima uscita in meno. Per esempio, chi si abbona a 12 numeri a partire dal n. 1/89 e richiede 5 arretrati, riceverà 7 numeri del 1989 e 5 arretrati.

Ritagliare e spedire a: IHT Gruppo Editoriale – Via Monte Napoleone, 9 – 20121 Milano

Sì, desidero sottoscrivere un abbonamento a ☐ 12 numeri (L. 81.600) ☐ 24 numeri (L. 163.200) di COMMODORE GAZETTE usufruendo così dello sconto del 15% sul prezzo di copertina

Nome e cognome ....

Indirizzo .... Città .... CAP ....

Inizio abbonamento dal n. .... Desidero ricevere i seguenti numeri arretrati (specificare numero e anno) ....

☐ Allego assegno bancario o circolare ☐ Allego fotocopia della ricevuta del vaglia postale

Firma ....

LE STAMPANTI OKI A 9 AGHI E A 18 AGHI

Due proposte OKI: la Microline 192/Elite, una stampante a 9 aghi con qualche soluzione originale, e la Microline 292/Elite, una stampante a 18 aghi che si fa apprezzare per la velocità e per la stampa a colori

di Luca Giachino

La OKI MICROLINE 192/Elite e la OKI MICROLINE 292/Elite sono due stampanti che appartengono a due fasce di prezzo molto vicine. La prima è a 9 aghi mentre la seconda a 18. Offrono buone prestazioni, soprattutto per quanto riguarda la definizione dei caratteri, ma appaiono esteticamente poco accattivanti, sicuramente non al passo con i design più in voga nell'affollato mondo delle periferiche. Anche sotto il profilo della solidità – stiamo pensando a quell'impressione piacevole che si prova quando si maneggia una periferica che non scricchiola e con plastica che non si flette – queste OKI non brillano particolarmente.

Internamente si presentano molto ordinate, con circuiti stampati che per integrazione non hanno niente da invidiare a quelli di altre stampanti della stessa fascia di prezzo. Nello sforzo di privilegiare le ridotte dimensioni non vengono però adottate soluzioni particolarmente felici per la meccanica di trascinamento e di stampa. Se costassero un po' meno, basterebbe la loro indubbia qualità di stampa a mettere in secondo piano gli aspetti meno riusciti. Ma nella valutazione del prezzo di una stampante non è possibile tenere conto soltanto dei buoni risultati di stampa. In un giudizio completo e obiettivo entrano sempre in gioco anche altri parametri (affidabilità, velocità...) che per alcuni utenti potrebbero essere persino più importanti della stessa qualità di stampa. Per quanto riguarda questa prestazione, infatti, il prezzo delle due OKI non appare affatto esagerato.

La OKI 192/Elite

In quest'ultimo periodo, la tecnologia della stampa a 9 aghi ha messo in mostra progressi davvero notevoli. Nonostante l'avvento di testine con un numero maggiore di aghi, i costi ridotti delle stampanti a 9 aghi e il progressivo aumento della loro velocità di stampa rimangono due elementi preferenziali per l'utente non professionale. Tanto più che anche la nitidezza dei caratteri è andata aumentando di pari passo con le prestazioni cronometriche.

Per quanto riguarda la velocità, i 240 cps (caratteri per secondo) a 12 cpi (caratteri per pollice), della OKI 192/Elite non rappresentano certo la punta di diamante della tecnologia attuale. Nella stessa fascia di prezzo esistono sul mercato stampanti a 9 aghi molto più veloci, meno rumorose e anche più user-friendly.

La OKI 192/Elite, d'altra parte, si fa apprezzare per le dimensioni molto ridotte (360 x 275 x 80 mm) che la rendono idonea a spazi di lavoro limitati senza pregiudicarne la funzionalità. Infatti, anche se minuta, la OKI 192/Elite non rinuncia a offrire prestazioni ormai consuete come i due trascinamenti tipici, il trattore e la frizione, e uno schienale guida ribaltabile per la carta, molto comodo per inserire i fogli singoli verticalmente. Sono anche disponibili gli incastri per inserire un più versatile trattore esterno, un supporto per moduli continui a rullo e un inseritore automatico per i fogli singoli (questi ultimi tre accessori sono tutti opzionali). Con una modica spesa aggiuntiva è possibile acquistare separatamente anche una particolare base d'appoggio che consente l'inserimento frontale dei fogli singoli. Nella base d'appoggio è presente una feritoia che corrisponde a un'uguale feritoia situata nella parte inferiore della stampante, in corrispondenza del rullo trasci-

natore. I fogli singoli, seguendo la guida interna della base d'appoggio, vengono così incanalati dal basso nella meccanica di trascinamento della stampante. Si tratta di un sistema manuale, e si potrebbe quindi ritenerlo una spesa inutile, ma per certi impieghi questo semplice inseritore di fogli singoli rappresenta il giusto compromesso fra la scomodità dell'inserimento verticale e il costo dell'inseritore automatico.

L'analisi esterna

La prima impressione di fronte alla OKI 192/Elite è quella di maneggiare una periferica d'impianto tradizionale, all'apparenza priva di elementi innovativi o che mettano in mostra almeno una certa personalità. Pesa soltanto 4 chilogrammi e mezzo, è di colore beige e non ha elementi sporgenti. Il pannello frontale, leggermente infossato, offre una console operativa formata da quattro tasti e tre indicatori a led. Non si può certo dire che i costruttori abbiano fatto grossi sforzi per rendere pratica l'interazione con la console. I tasti sono infatti poco sensibili, traballanti e fastidiosi da usare. I led, di vecchio modello, diventano praticamente invisibili con la luce diretta. Sono disponibili i controlli di line feed e form feed, top of form per indicare alla stampante l'inizio di un foglio di stampa, e select per passare da on-line a off-line e viceversa. Se all'accensione si tiene premuta una particolare combinazione di tasti si può abilitare il modo esadecimale di stampa, molto utile per controllare quali codici giungono effettivamente alla stampante. Un'altra combinazione dà il via all'autotest, e un'altra ancora mette in azione il programma d'intervento sugli switch software. Quest'ultima possibilità, disponibile in tutti i modelli OKI, permette di configurare permanentemente la stampante senza dover accedere a remoti microswitch interni, ma obbliga a inserire un foglio di stampa per vedere in sequenza le opzioni che si possono modificare.

Sul lato superiore notiamo lo sportello che permette un facile accesso alla meccanica di stampa per sostituire il caricatore d'inchiostro (molto piccolo, come vedremo, ma di lunga durata). Purtroppo lo sportello a incastro non si toglie facilmente ed è fatto di una plastica "trasparente" fumé talmente scura che in pratica l'utente è obbligato ad aprirlo sempre, se vuole leggere la riga stampata senza sforzare troppo la vista.

Il rullo trascinatore è quasi completamente nascosto dallo sportello. Sul suo asse di rotazione, in corrispondenza dei due estremi, sono montate direttamente le ruote dentate che agganciano i moduli perforati. È una scelta dettata dall'esigenza di ridurre l'ingombro al minimo, ma a nostro avviso si tratta comunque di una scelta poco felice. Primo, perché rende difficile l'aggancio del foglio perforato. Non è possibile infatti disporre manualmente i fori sui denti delle ruote trascinatrici, e si è costretti a dare il line feed senza nessuna certezza che il foglio venga agganciato correttamente. Secondo, perché questo sistema permette un'escursione orizzontale minima delle due ruote dentate (appena un paio di centimetri) e quindi non consente di trascinare moduli perforati più stretti del normale, come per esempio quelli per etichette. Al primo problema forse ci si può abituare, ma per il secondo l'unico rimedio economico è utilizzare solo il trascinamento a frizione, con tutti i problemi che comporta la frizione quando si usano i moduli continui. Esiste però un trattore esterno a incastro, che si può acquistare separatamente, in grado di ricevere fogli perforati di qualsiasi larghezza.

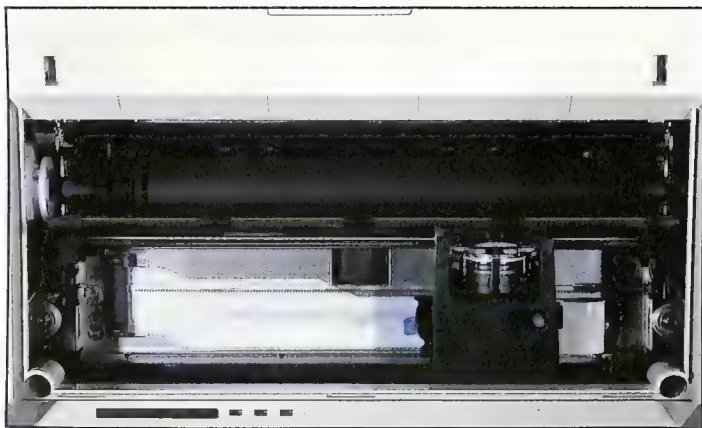
Alla sinistra del rullo di stampa è presente la leva che distanzia il premi-carta, mentre a destra è presente la leva che permette di selezionare il tipo di trascinamento (a trattore o a frizione). Appena dietro il rullo sono visibili gli incastri che possono ricevere lo schienale guida ribaltabile per la carta (di serie), il supporto per i moduli continui a rullo, l'inseritore di fogli singoli automatico. Dietro si trova uno sportellino che ha l'unica funzione di permettere l'accesso ai microswitch dell'eventuale interfaccia seriale. Lo schienale ribaltabile possiede una guida scorrevole per inserire i fogli singoli sempre allo stesso margine sinistro; peccato che ce ne sia soltanto una.

Sul retro della stampante troviamo a destra il connettore dell'interfaccia parallela Centronics (di serie) e lo sportello in plastica che dev'essere tolto se si monta anche l'interfaccia seriale (opzionale). Al centro c'è una piccola presa per collegare l'inseritore di fogli singoli opzionale, e sulla sinistra la presa per il cavo dell'alimentazione. Va notata l'infelice posizione del connettore per l'interfaccia, peraltro comune a moltissime stampanti, che, a cavo collegato, interferisce con il percorso del modulo continuo. Come vedremo, nella OKI 292/Elite questo connettore risiede sul lato sinistro del cabinet, una soluzione che rende più agevole lo scorrimento del modulo. L'interruttore si trova sul lato sinistro e risulta quindi di facile accesso.

Di solito la base di una stampante non presenta caratteristiche degne di nota, ma questa volta la OKI 192/Elite costituisce un'eccezione. Come avevamo anticipato, vi si trova la feritoia che permette ai fogli singoli di essere inseriti nel meccanismo di trascinamento dal basso, una possibilità utilizzabile quando la stampante si trova nella configurazione per l'inserimento manuale di fogli singoli. Comunque, questa via per l'inserimento della carta si può usare anche per i moduli perforati, e senza l'inseritore manuale.

Nel complesso l'analisi esterna di questo prodotto, a parte l'ultima caratteristica che abbiamo descritto, non denota particolari di rilievo, soluzioni inedite o idee brillanti. Si tratta della solita veste che potremmo aspettarci dalla generica stampante di poche pretese.

Le caratteristiche

Iniziamo dalla velocità di stampa. Nella stampa bidirezionale la OKI 192/Elite consente due diverse velocità, e quindi due tipi di qualità. In qualità Draft si ottengono 240 cps (a 12 cpi), mentre attivando la qualità Utility la velocità scende a 160 cps (a 10 cpi). La differenza fra i due modi di stampa è solo la maggiore densità di punti del secondo (9 x 9 contro 7 x 9). La velocità è in linea con la media delle stampanti a 9 aghi. Se invece si abilita il modo Near Letter Quality la velocità scende a 40 cps e i caratteri appaiono molto più marcati (17 x 17).

Si notano il caricatore d'inchiostro e il sistema di trascinamento della testina

Le spaziature disponibili sono 5, 6, 8,5, 10, 12, 17,1, e 20 cpi. Sulla riga di stampa possono quindi apparire da un minimo di 40 caratteri, a un massimo di 160 (a 5 cpi i caratteri però sono larghi il doppio). È anche possibile abilitare la spaziatura proporzionale, grazie alla quale si simula il comportamento dei caratteri tipografici.

Per quanto riguarda la lunghezza del foglio di stampa, la OKI 192/Elite permette di scegliere fra 11 diverse dimensioni, da 14 pollici a 3 pollici. Questa varietà è senz'altro degna di nota, e si rivela idonea per stampare fatture, ricevute, e tutto quel genere di modulistica che richiede fogli di lunghezza non standard.

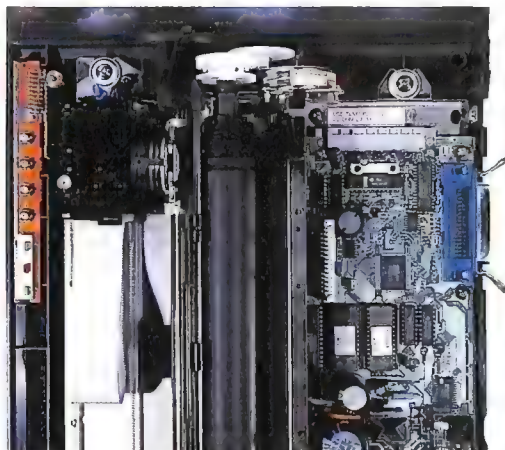
Oltre alle spaziature e alla lunghezza del foglio, è ovviamente possibile variare lo stile del carattere. Alcune variazioni di stile non vengono riconosciute con tutte le opzioni di stampa. In particolare, notiamo che nella qualità Utility abbiamo il maggior ventaglio di possibilità. Gli stili disponibili sono i consueti tondo, corsivo, nero, neretto, apici e pedici.

La compatibilità della OKI 192/Elite con gli standard Epson FX 85/105, IBM Proprinter XL e IBM Graphics Printer garantisce la presenza di tutta una serie di comandi ormai comuni alla maggioranza delle stampanti in commercio. Per l'Amiga, in particolare, lo standard Epson assicura sempre un perfetto interfacciamento. Per selezionare l'emulazione, come avviene con tutti gli altri parametri di configurazione permanente, si agisce attraverso l'apposito programma presente nel firmware. Inserendo un foglio e accendendo la stampante, mentre si tiene premuta un'opportuna combinazione di tasti, il programma riceve il controllo e procede a interrogare l'utente sui seguenti parametri:

- la qualità di stampa all'accensione (Utility, Draft, NLQ o Italics)
- la spaziatura
- l'interlinea
- lo stile
- il set di caratteri (da scegliere fra due residenti e uno caricabile via software)
- il set nazionale dei simboli
- la stampa grafica monodirezionale (più precisa di quella bidirezionale, ma più lenta)

- la lunghezza del foglio di stampa
- il salto della perforazione
- la generazione automatica dei caratteri di line feed e di carriage return
- l'attivazione del sensore di fine carta
- i tipi di emulazione
- lo zero sbarrato
- l'aggancio e la collocazione automatica del foglio singolo alla prima riga di stampa, e altri di minore importanza.

Come si vede, i parametri su cui si può intervenire sono numerosi e il programma per modificarli è davvero semplice da usare.

L'ultima caratteristica che ci pare degna di nota è un sistema di rilevamento termico applicato alla testina, che blocca il funzionamento della stampante quando la temperatura della testina supera una certa soglia. Il costruttore dichiara che questo sistema è stato adottato per diminuire l'usura a lungo termine della testina. Non si tratta quindi di un sistema per sopperire all'inefficienza della testina, ma di un modo per evitare stress eccessivi che nel tempo possono a poco a poco peggiorarne le prestazioni. Ci sembra un sistema di autocontrollo interessante, soprattutto se la stampante si trova a lavorare per ore e ore in ambienti molto caldi, magari d'estate. Nella nostra prova la stampante non si è mai bloccata ma, data la stagione e la temperatura esterna, non c'è da stupirsene troppo.

Sulla destra il circuito di controllo con l'interfaccia parallela

La prova

Nella prova la OKI 192/Elite ci ha riservato una piacevole sorpresa. Nonostante l'apparenza poco accattivante, ha messo in mostra una meccanica in grado di fornire stampati di buona qualità. I caratteri appaiono ben marcati,

Prova di stampa della
OKI Microline 192 Elite

Qualita' di stampa disponibili

Draft	Utility	Near Letter Quality
abcdefghijklm	abcdefghijklm	abcdefghijklm
NOPQRSTUVWXYZ	NOPQRSTUVWXYZ	NOPQRSTUVWXYZ
1234567890+-*/	1234567890+-*/	1234567890+-*/
,.;:@#!$%&()=?^	,.;:@#!$%&()=?^	,.;:@#!$%&()=?^

Stili di stampa disponibili

Draft	Utility	Near Letter Quality
---	*Corsivo*	---
---	Nero	Nero
---	**Neretto**	---
---	Proporzionale	Proporzionale
Apici	Apici	Apici
Pedici	Pedici	Pedici

Dimensioni particolari del carattere

Larghezza doppia

precisi nei dettagli e negli allineamenti verticali (soprattutto in NLQ). Peccato che la stampa non sia un po' più rapida e silenziosa.

Su altri fronti, però, la OKI 192/Elite perde terreno. Con il tempo si finisce sempre per abituarsi ai grandi problemi, mentre quelli meno importanti finiscono per diventare i più fastidiosi. Per questo la praticità di una stampante è una caratteristica tanto importante e si spera sempre che il costruttore non abbia pensato solo alle prestazioni, ma anche alla comodità dell'utente. Per questa stampante, la casa produttrice ha voluto privilegiare le piccole dimensioni, lasciando la possibilità di migliorare certi aspetti poco felici con opportuni accessori opzionali. Ci riferiamo per esempio al sistema di trascinamento con i trattori direttamente montati sul rullo di stampa. Un'idea a nostro avviso davvero infelice, che serve soltanto a irritare l'utente ogni volta che deve passare dalla frizione al trascinamento a trattore e viceversa. Si nota ancora la scadente risposta dei tasti di controllo della console e la conseguente impressione di precarietà.

Un'altra soluzione poco felice è l'aver collocato il sensore di fine carta molto vicino al limite sinistro del rullo di stampa. Questa posizione così angolata fa sì che talvolta, inserendo un foglio senza accostarlo a sinistra, la stampante rimanga in stato di allarme, con ovvio stupore dell'utente. È vero che il sensore di fine-carta si può disabilitare, ma se fosse stato collocato in una posizione più centrale, non ne avremmo neanche parlato.

La OKI 192/Elite è accompagnata da un manuale dettagliato e corposo: né l'utente né il programmatore potrebbero desiderare una fonte d'informazioni più completa. L'impiego dei comandi più importanti viene illustrato anche per mezzo di alcuni semplici esempi in Basic, di facile comprensione per chiunque abbia un po' di familiarità con la programmazione.

L'analisi interna

Aprendo la OKI 192/Elite, si nota subito che a monte di questo prodotto esiste un grande nome di livello mondiale. Innanzitutto, il circuito elettronico che governa la stampante è molto curato, e tecnologicamente al passo coi tempi. Molti componenti, infatti, sono saldati direttamente sulle piste di rame (tecnologia SMD, Surface Mount Devices), risparmiando lo spazio dei piedini. Su questo circuito risiede la ROM del firmware di gestione della stampante, inserita in uno zoccolo, e l'interfaccia parallela Centronics. Se si acquista anche l'interfaccia seriale, occorre montarla su appositi distanziatori disposti intorno alla parte del circuito che costituisce l'interfaccia parallela, e collegarla tramite un opportuno connettore. Si tratta di un'operazione relativamente semplice, che non richiede particolare esperienza.

L'interfaccia seriale, denominata *Super Speed RS-232C*, è molto sofisticata; permette infatti di comunicare a varie velocità, fino a un massimo di 19.200 baud, e invia un segnale d'avvertimento se i parametri impostati tramite i microswitch non sono compatibili con il formato dei dati in arrivo. Infine, accanto al circuito di controllo della stampante, risiede quello dell'alimentatore. Anche questo è ben strutturato, senza elementi posticci.

La meccanica di avanzamento della carta lascia qualche dubbio

DATI TECNICI DELLA OKI 192/ELITE

Testina di stampa:
9 aghi a impatto

Velocità di stampa:
240 cps Draft a 12 cpi
160 cps Utility a 10 cpi
40 cps NLQ a 10 cpi

Densità di stampa:
5, 6, 8,5, 10, 12, 17,1, 20 cpi

Caratteri per linea:
80 cpl
96 cpl
132 cpl
160 cpl

Trascinamento:
A trattore per moduli perforati
A frizione per fogli singoli
Inserimento carta anche dal basso

Emulazioni:
Epson FX
IBM Graphics Printer
IBM Proprinter

Interfaccia di serie:
Parallela Centronics

Affidabilità:
Vita della testina:
200 milioni di caratteri
MTBF (Tempo medio tra un guasto e il successivo): 4000 ore

Vita del caricatore d'inchiostro:
3 milioni di caratteri

Rumorosità:
55 decibel

Dimensioni:
360 x 275 x 80 mm

Peso:
4,5 Kg

Prezzo al pubblico:
L. 1.020.000 + IVA (parallela)
L. 1.205.000 + IVA (seriale)

Optional

Interfaccia seriale RS-232C o 422:
L. 220.000 + IVA

Caricatore manuale di fogli singoli:
L. 65.000 + IVA

Caricatore automatico di fogli singoli:
L. 430.000 + IVA

Trattore esterno a incastro:
L. 100.000 + IVA

Supporto per moduli continui a rullo:
L. 80.000 + IVA

Produzione:
OKI Electric Industry Co. Ltd.
Tokyo, Giappone

Distribuzione:
Technitron Data S.p.A.
Centro Commerciale Il Girasole
Lotto 3.05/B
Lacchiarella, Milano
(tel. 02/90076410)

per l'abbondanza di elementi in plastica. Si nota che questa parte della stampante non raggiunge lo stesso livello di finitura della parte elettronica. La stessa testina è montata su un unico albero, anziché due come normalmente avviene, il che potrebbe causare una non perfetta stabilità nel movimento tangenziale della testina. Inoltre, il carrello che sorregge la testina si muove grazie a un ingranaggio che fa leva su di un'asta dentellata (anch'essa di plastica) disposta lungo tutta la larghezza del rullo di stampa; rispetto a questo sistema di trascinamento della testina, è preferibile quello a cinghia, comune alla maggior parte delle stampanti. Troviamo invece interessante la soluzione adottata per il caricatore del nastro inchiostrato. Oltre a essere montato direttamente sul carrello della testina, è infatti di dimensioni molto ridotte. Si potrebbe pensare che questo pregiudichi la sua durata nel tempo, ma sia i dati del costruttore sia la nostra prova confermano che invece questo caricatore ha una vita pari a quella dei suoi fratelli maggiori. Infine, non è positivo che la testina si muova così facilmente quando si solleva la stampante. In pratica, per qualsiasi semplice operazione di spostamento della stampante, magari da una scrivania a un'altra, se non si provvede a bloccare la vivace testina, si rischia di danneggiarla.

Conclusioni

La OKI 192/Elite non ci ha affascinati. In confronto alla media delle sue dirette concorrenti non introduce caratteristiche di rilievo, e anzi su alcuni fronti si trova perfino in svantaggio. Anche se stampa egregiamente, a un livello che non ci saremmo aspettati, non possiamo dimenticare le sue carenze (soprattutto a causa di un prezzo che non permette indulgenze di alcun tipo). In definitiva, giudichiamo questa stampante un prodotto mediocre, con qualche soluzione interessante, come l'inserimento della carta dal basso, ma fondamentalmente priva di reali attrattive.

Rimane da sottolineare l'unico vero grande pregio: le sue ridotte dimensioni, che possono trovare il favore di coloro che non intendono affollare troppo il proprio studio.

La OKI 292/Elite

A differenza della OKI 192/Elite, questa stampante è dotata di una testina a 18 aghi. Questo ovviamente si traduce in una matrice verticale di stampa a maggiore densità di punti per unità di lunghezza, cioè in caratteri molto più pieni. I risultati che si ottengono in stampa la avvicinano più alle testine a 24 aghi (molto più comuni) che a quelle a 9 aghi. Certo, le stampanti a 18 aghi sono poco diffuse, ma probabilmente il motivo sta solo nel desiderio d'investire il proprio denaro nel salto qualitativo più vantaggioso, quando si cambia stampante... e superando i 24 aghi i costi s'impennano. La OKI 292/Elite si presenta meglio del modello inferiore, con un design leggermente più efficace. Anch'essa è di dimensioni ridotte (367 x 305 x 105 mm), e quindi ricalca la filosofia di mettere al primo posto la riduzione dell'ingombro a costo di pregiudicare parzialmente la funzionalità. Non a caso, infatti, la OKI rende disponibile per questa stampante, come per la 192, diversi accessori che evidentemente aumentano l'ingombro, ma anche la funzionalità.

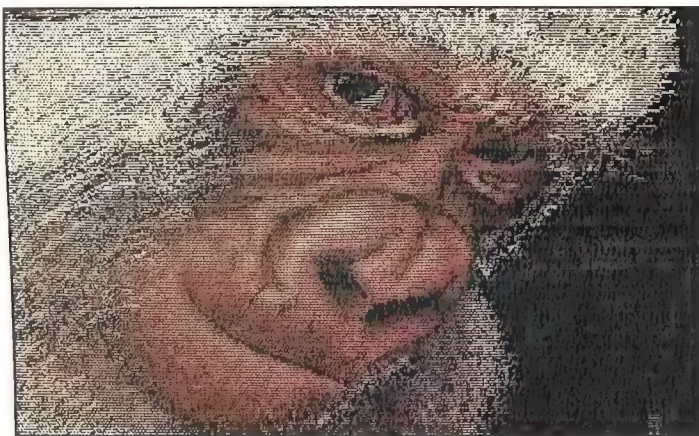
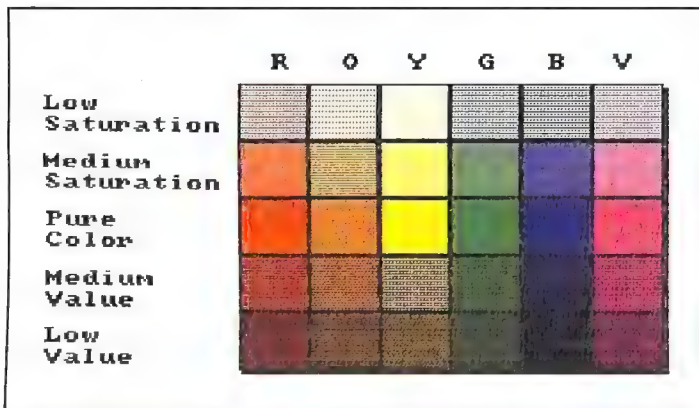
La OKI 292/Elite è dotata tanto del trascinamento a trattore quanto di quello a frizione e possiede uno schienale guida ribaltabile per la carta. Inoltre, con l'opportuno caricatore d'inchiostro è in grado di stampare a colori (per la gioia di chi sfrutta la grafica dell'Amiga). La sua massima velocità di stampa è di 300 cps (a 12 cpi) ed è un po' rumorosa. Gli accessori opzionali sono il trattore esterno a incastro, l'inseritore automatico di fogli singoli, e la base d'appoggio che permette l'inserimento frontale dei fogli singoli. Dal momento che molte caratteristiche sono comuni a entrambi i modelli, quelle già descritte per la 192 verranno solo accennate.

L'analisi esterna

Il lato frontale contiene la console di controllo, praticamente identica a quella della 192 e dunque neppure il nostro parere subisce modifiche. Anche per questa stampante, tramite la con-

sole è possibile attivare l'autotest, la stampa esadecimale dei codici in arrivo all'interfaccia e l'attivazione di un programma di modifica della configurazione permanente, che rispetto a quello della 192 aggiunge solo alcuni parametri. Sul lato superiore l'unica differenza con la 192 è uno sportello più facile da rimuovere. Anche qui si notano gli incastri per ricevere i vari accessori, ma manca lo sportello per accedere ai microswitch dell'eventuale interfaccia seriale, in quanto su questo modello l'interfaccia è un compatto modulo estraibile. Lo schienale guida della carta è invece un po' più sofisticato, dal momento che mette a disposizione una barra metallica nella quale si può far passare il modulo continuo quando lo schienale non è sollevato. In questo modo la carta scivola sulla barra senza strisciare sul cabinet. In tale posizione lo schienale funge anche da separatore fra la carta in arrivo e quella in uscita.

Sulla destra della stampante è presente l'interruttore (a cui si può accedere senza contorsioni) mentre sul retro ci sono solo la presa per il collegamento alla rete elettrica, e una presa di controllo per l'eventuale inseritore automatico di fogli singoli. Sul lato sinistro compare il connettore dell'interfaccia in uso. A questo proposito osserviamo che, con questa posizione del connettore, il cavo di collegamento al computer non interferisce assolutamente col percorso della carta, anche se viene un po' aumentato l'ingombro laterale della stampante. Inoltre, è senz'altro positiva la scelta di adottare per la OKI 292/Elite un compatto modulo d'interfaccia estraibile, che permette una semplicissima rimozione qualora si desideri cambiare il protocollo di trasmissione con il computer. Infine, la OKI 292/Elite, come il modello a 9 aghi, presenta sul fondo la feritoia per l'accesso dal basso del foglio di stampa.

Per quanto riguarda la meccanica di trascinamento, anche su questo modello, per le solite ragioni di spazio, i trattori sono montati direttamente sull'asse di rotazione del rullo di stampa, con gli stessi problemi (e le stesse soluzioni) che abbiamo già visto per la 192.

Sopra: i colori generati dalla 292/Elite sono ben marcati in tutte le tonalità. Sotto: la prova di stampa a colori mostra la buona uniformità dell'immagine

Le caratteristiche

Innanzitutto, la OKI 292/Elite prevede quattro qualità di stampa. In modo Draft raggiunge i 300 cps (a 12 cpi). A questa ragguardevole velocità i caratteri che si ottengono sono costituiti da una matrice di 7 x 9 aghi, e quindi non sono particolarmente belli, ma nel caso di veloci riscontri su carta possono andare benissimo. In qualità Utility (matrice del carattere 9 x 9) la velocità scende a 200 cps (a 10 cpi), ma il tipo di carattere non migliora sensibilmente rispetto alla qualità Draft. Per osservare apprezzabili miglioramenti, è necessario attivare una delle due qualità Near Letter Quality (matrice del carattere 17 x 17). Con tutti i 18 aghi a disposizione, i caratteri appaiono finalmente molto ben definiti e la velocità di stampa non scende drasticamente come ci si potrebbe aspettare, attestandosi sui 100 cps. Si tratta di un dato importante, perché supera di

gran lunga la media. In NLQ si può praticamente stampare alla velocità che sulle stampanti a basso costo si ottiene soltanto in modo Draft. Quindi, sotto il profilo della qualità del carattere, e delle velocità disponibili, la OKI 292/Elite ci ha favorevolmente impressionato.

Le spaziature variano, come per la 192, da 5 a 20 cpi, offrendo così una vasta scelta in grado di soddisfare molteplici esigenze, soprattutto nel caso della stampa su moduli predefiniti. Sempre in riferimento a particolari esigenze, la lunghezza del foglio di stampa può essere selezionata fra ben 12 valori, da 3 a 14 pollici. Peccato però che con la dotazione di serie, come con la 192, non sia possibile stampare su fogli di larghezza inferiore agli 8,5 pollici standard.

Per quanto riguarda gli stili disponibili, valgono le considerazioni fatte per il modello inferiore, tenendo presente che anche per la OKI 292/Elite il maggior ventaglio è disponibile in qualità Utility. Per quanto riguarda invece le dimensioni del carattere, questa stampante aggiunge la possibilità di raddoppiare l'altezza dei caratteri. In questo modo, soprattutto se si raddoppia anche la larghezza, si può stampare un testo molto più leggibile. Peccato solo che raddoppiando l'altezza, ogni carattere viene completato con due passate della testina su due righe, e i risultati mettono in luce una certa irregolarità negli allineamenti verticali, probabilmente a causa di scelte meccaniche non troppo felici.

Intorno alla testina si nota il caricatore del nastro a colori

Come per la 192, è assicurata la compatibilità con gli standard più diffusi (Epson FX/EX, IBM Proprinter XL e Graphics Printer), e l'utente dell'Amiga non incontrerà problemi selezionando il driver di

Prova di stampa della
OKI Microline 292 *Elite*

Qualita' di stampa disponibili

Qualita' Draft
abcdefghijk LMNOPQRSTUVWXYZ
1234567890 +-.,;:#=)(/%$@!?

Qualita' Utility
abcdefghijk LMNOPQRSTUVWXYZ
1234567890 +-.,;:#=)(/%$@!?

Qualita' NLQ Courier
abcdefghijk LMNOPQRSTUVWXYZ
1234567890 +-.,;:#=)(/%$@!?

Qualita' NLQ Sans Serif
abcdefghijk LMNOPQRSTUVWXYZ
1234567890 +-.,;:#=)(/%$@!?

Stili di stampa disponibili

Draft	Utility	NLQ Courier	NLQ Sans Serif
... ...	*Corsivo*	*Corsivo*	*Corsivo*
... ...	Nero	Nero	Nero
... ...	Neretto	---	---
... ...	Proporzionale	Proporzionale	Proporzionale
Apici	Apici	Apici	Apici
Pedici	Pedici	Pedici	Pedici

Dimensioni particolari del carattere

Altezza doppia

Larghezza doppia

Altezza e larghezza doppia

stampa Epson. Lo standard da emulare, insieme con tutti gli altri parametri che costituiscono la configurazione permanente della stampante, è selezionabile tramite l'apposito programma presente nel firmware. Rispetto ai parametri visti con la 192, la OKI 292/Elite aggiunge il controllo dell'altezza del carattere (normale o doppia), la selezione del colore dell'inchiostro da usare all'accensione (quando è presente il caricatore a colori), e la selezione di un maggior numero di set di caratteri non residenti, cioè caricabili via software. La gestione di questo programma è la stessa prevista dalla 192.

Prima di procedere con la prova, ricordiamo infine che anche questa stampante dispone sulla testina del sensore termico per bloccare la stampa in caso di riscaldamento eccessivo.

Si notino il modulo d'interfaccia in alto a sinistra e l'ordine del cablaggio

La prova e il colore

Sul banco della verità, la OKI 292/Elite non ci ha delusi. Proprio perché in termini di prestazioni non sono tante le concorrenti a 18 aghi con cui confrontarla, i dati ottenuti li abbiamo accostati alla media che si riscontra nella famiglia delle stampanti a 24 aghi, e questo ci ha permesso di apprezzare in modo particolare le velocità di stampa disponibili (soprattutto i 300 cps in qualità Draft e i 100 cps in qualità NLQ). Infatti queste due prestazioni di buon livello ci consentono di segnalare la OKI 292/Elite come una stampante molto versatile, efficiente su due fronti spesso incompatibili: la velocità e la qualità. Specialmente i 100 cps nelle due qualità NLQ ci sembrano un dato di assoluto rilievo.

La qualità di stampa si può ritenere buona solo in NLQ, mentre nelle qualità Draft e Utility i risultati sono nettamente inferiori. Peccato per questo drastico divario, sarebbe stata preferibile una qualità di stampa intermedia, anziché due molto simili come la Draft e la Utility. Inoltre, soltanto in NLQ viene sfruttata l'alta densità di punti possibile con la testina a 18 aghi, mentre nelle altre qualità il comportamento è quello di una normale 9 aghi.

Sebbene la OKI 292/Elite mostri un buon comportamento nella stampa dei caratteri, lo stesso non si può dire degli allineamenti verticali quando è attiva la stampa bidirezionale, come si può notare in modo particolare stampando i caratteri di altezza doppia. La situazione migliora, come c'era da aspettarsi, se si attiva la stampa monodirezionale.

Come abbiamo già accennato, la OKI 292/Elite, senza aggiungere alcun kit alla dotazione di serie, è in grado di stampare a colori. Dotandola di un apposito caricatore a colori, uno speciale sistema meccanico provvede a muovere il nastro verso l'alto e verso il basso per selezionare il colore da imprimere sulla carta. I risultati che si ottengono sono soddisfacenti, e ci inducono a pensare che la OKI 292/Elite possa costituire un prodotto molto interessante per coloro che s'interessano di grafica. L'unico neo riguarda il firmware di gestione, che non esegue alcun controllo sul buffer di riga, il che si traduce in fastidiose perdite di tempo quando la riga di stampa non contiene niente da stampare, e la 292 continua imperterrita a muovere per tre volte la testina. Per non parlare dello spreco di nastro.

Per quanto riguarda le impressioni d'uso, valgono le stesse considerazioni fatte per la 192: un sistema di trascinamento a trattore che non agevola l'inserimento del foglio perforato, la posizione troppo angolata del sensore di fine carta, un controllo da console che non si rivela molto agevole e non suscita quell'impressione di solidità e precisione che si vorrebbe. Inoltre, come il modello inferiore, anche la 292/Elite appare un po' rumorosa.

Rendiamo onore, infine, a una documentazione davvero ottima. Con la OKI 292/Elite vengono infatti forniti due manuali, uno per l'installazione, con tanto di foto a colori, e uno per l'uso della stampante, sovrabbondante di dettagli sia per l'utente sia per il programmatore. Questo aspetto merita di essere sottolineato, perché non capita spesso di trovare documentazioni davvero complete insieme alle stampanti. Oltre ai due manuali, la OKI correda la stampante anche di dischi da 5,25" contenenti dati e utility in

formato MS-DOS. I programmi più interessanti sono un Font editor che permette di creare fonti-carattere personalizzate, e un'applicazione che consente di scegliere una delle fonti presenti sul disco e d'inviarla alla stampante (sul disco sono contenute diverse fonti già pronte per l'uso).

L'analisi interna

Aprendo il cabinet, la 292/Elite rivela un cablaggio della meccanica e dell'elettronica pulito e ordinato. In particolare si nota il circuito di controllo elettronico, sul quale risiede anche la ROM del firmware montata su uno zoccolo. Riscontriamo anche per questo circuito il buon livello d'integrazione e l'abbondanza di componenti di tipo SMD, che conferiscono all'insieme un aspetto professionale. Sulla destra del circuito di controllo spicca un voluminoso e sofisticato alimentatore, anch'esso ben congegnato e ordinato. Sulla sinistra invece appare il compatto modulo d'interfaccia estraibile. Si tratta di un buon sistema, che facilita notevolmente la sostituzione dell'interfaccia, e garantisce la compatibilità con gli standard di comunicazione che potranno affermarsi in futuro.

Per quanto riguarda la meccanica, dobbiamo ribadire quanto è stato osservato per la 192. Vi sono alcuni aspetti negativi, come l'abbondanza di plastica, la presenza di un solo vincolo di scorrimento della testina, e un sistema di trascinamento della testina che prevede, come per la 192, un'asta dentellata in plastica anziché la cinghia impiegata dalla maggior parte delle stampanti. Forse è questa la ragione dei non perfetti allineamenti verticali che abbiamo notato nella prova. Inoltre, il carrello che sorregge la testina si dimostra anch'esso troppo sensibile alle inclinazioni della stampante e, se non viene bloccato, finisce per danneggiarsi anche durante il più banale movimento della stampante. Il caricatore d'inchiostro, a differenza di quello della 192, è fisso e copre l'intera larghezza della riga di stampa. La sostituzione è agevole, ma ci è sembrato complicato il meccanismo che lo muove per selezionare i colori. È anche vero che sotto questo profilo se ne vedono davvero... di tutti i colori, ma il sistema adottato dalla OKI 292/Elite ci sembra uno dei peggiori, soprattutto per l'eccessiva abbondanza d'ingranaggi. Comunque, per quanto complicato, nella prova non ci ha dato motivo di sospettare rischi di malfunzionamenti.

Conclusioni

Terminiamo questa prova con una considerazione già emersa dai paragrafi precedenti: la OKI 292/Elite merita un giudizio più favorevole della OKI 192/Elite, soprattutto perché il suo prezzo non si discosta molto da quello del modello a 9 aghi, ma le prestazioni sono di gran lunga superiori, in termini di velocità di stampa e densità del carattere. Inoltre dà la possibilità di stampare anche a colori, un aspetto che in ambito Commodore è molto importante.

Se abbiamo definito sproporzionato il prezzo della 192, quello della 292 ci appare già più ragionevole. Certo, neanche questo modello è esente da difetti, ma le sue qualità riescono a farli pesare meno sul piatto della bilancia. E l'ago si sposta così verso una valutazione abbastanza positiva, riabilitando le tecnologia di stampa a 18 aghi, il compromesso incompreso. ■

DATI TECNICI DELLA OKI 292/ELITE

Testina di stampa:
18 aghi a impatto

Velocità di stampa:
300 cps Draft a 12 cpi
200 cps Utility a 10 cpi
100 cps NLQ a 10 cpi

Densità di stampa:
5, 6, 8,5, 10, 12, 17,1, 20 cpi

Caratteri per linea:
80 cpl
96 cpl
132 cpl
160 cpl

Buffer di stampa:
12K

Trascinamento:
A trattore per modulo perforato
A frizione per fogli singoli
Inserimento carta anche dal basso

Emulazioni:
Epson FX
IBM Graphics Printer
IBM Proprinter

Interfaccia di serie:
Parallela Centronics

Affidabilità:
Vita della testina:
200 milioni di caratteri
MTBF (Tempo medio tra un guasto e il successivo): 4000 ore

Vita del caricatore d'inchiostro:
3 milioni di caratteri

Rumorosità:
55 decibel

Dimensioni:
367 x 305 x 105 mm

Peso:
5,7 Kg

Prezzo al pubblico:
L. 1.300.000 + IVA
(parallela o seriale)

Optional

Interfaccia seriale:
L. 320.000 + IVA

Caricatore manuale di fogli singoli:
L. 65.000 + IVA

Caricatore automatico di fogli singoli:
L. 450.000 + IVA

Trattore esterno a incastro:
L. 125.000 + IVA

Produzione:
OKI Electric Industry Co. Ltd.
Tokyo, Giappone

Distribuzione:
Technitron Data S.p.A.
Centro Commerciale Il Girasole
Lotto 3.05/B
Lacchiarella, Milano
(tel. 02/90076410)

DAL NOSTRO INVIATO A FRANCOFORTE

LA PAROLA AI PROGRAMMATORI: STA NASCENDO L'AMIGA 3000?

Le nuove versioni di Kickstart e Workbench, le schede A2620 e A2286 e l'Enhanced Chip Set stanno forse preparando l'Amiga del futuro. Ma la Commodore annuncia ben poche novità rispetto a sei mesi fa

di Michael Moosleitner

Si è tenuta a Francoforte la seconda "Amiga Developers Conference" europea. Dal 15 al 18 gennaio l'immenso hotel Intercontinental è stato il punto d'incontro per più di 300 programmatori e progettisti di hardware giunti da tutto il mondo.

Oltre 20 esperti del mondo Amiga sono intervenuti nelle varie sessioni e hanno risposto direttamente ai quesiti dei partecipanti. Tra i nomi più celebri ricordiamo Carolyn Scheppner e Andy Finkel, che guidavano il gruppo dei programmatori di West Chester; John Campbell e la signora Gail Wellington, responsabili del supporto ai prodotti Commodore (il primo si è rivelato un insuperabile moderatore). John Toebes VIII, artefice delle versioni per l'Amiga dei compilatori *Lattice*, è venuto dagli Stati Uniti con una valigia piena di programmi. C'era anche Guy Wright, il simpatico creatore della rivista *AmigaWorld*.

Una nota di merito spetta a Gunda O'Neal, che da Francoforte ha organizzato materialmente la conferenza (facendo i conti quotidianamente con la pessima qualità dei servizi messi a disposizione dall'albergo), e al viennese Thomas Giger, che si è dato da fare perché a nessuno mancasse l'hardware su cui lavorare.

Gli argomenti su cui si è dibattuto (nonché la documentazione consegnata ai partecipanti) sono stati praticamente gli stessi della conferenza di Washington del maggio scorso. Una piccola delusione, quindi, per chi ha pagato una seconda volta del materiale già in suo possesso. Molti programmatori hanno così preferito utilizzare il tempo delle sessioni mattutine per recuperare il sonno perduto di notte, piuttosto che ascoltare ripetizioni di cose già ben note. Non sono però mancate le novità, soprattutto per chi ha avuto la possibilità d'intrattenersi personalmente con i vari esperti.

Sono state discusse le varie caratteristiche delle versioni 1.4 e 1.5 del sistema operativo. Andy Finkel ha fatto scoppiare un lungo applauso quando ha annunciato che sono stati finalmente eliminati tutti i "Guru Meditation". Forse avrebbe fatto meglio a precisare che si tratta di un semplice maquillage, e che i Guru si chiameranno "System Messages". Lo stesso Finkel ha fatto poi sorridere il pubblico descrivendo gli stratagemmi adottati per far sembrare più veloce l'interfaccia utente della versione 1.4. Non ci sarà più bisogno neppure dei "Bug Fixes" (correzioni di errori). Anche per questi la Commodore ha infatti trovato un nuovo nome: "Feature Enhancements" (miglioramenti).

Lo scopo principale della versione 1.4, comunque, è di dare pieno supporto ai nuovi chip del set Enhanced. Jim Mackraz ci ha fatto assistere a un'eccitante anteprima dietro le quinte del nuovo sistema operativo e abbiamo scoperto, tra l'altro, che il puntatore del mouse sarà finalmente proporzionale: per muoversi da un estremo all'altro di uno schermo 1008 x 1024 non sarà più indispensabile un tavolo da riunioni. Basterà muovere il mouse un po' più velocemente. Gli schermi potranno essere spostati anche orizzontalmente e sarà possibile lavorare con i "public screens", cioè schermi diversi da quello standard del *Workbench*, ma condivisibili da più programmi. Grazie ai nuovi chip, inoltre, sarà possibile aprire uno schermo in "Productivity Mode", cioè 640 x 480 (con quattro colori scelti tra 64) in modo non interlace, a patto di possedere un monitor multisync o bi-sync. Questa nuova risoluzione sarà comune agli ambienti PAL e

NTSC. Le diverse risoluzioni video e gli stessi modi PAL, NTSC e VGA (sigla spesso utilizzata dai programmatori della Commodore quando parlano del Productivity Mode) potranno essere attivati grazie ai nuovi chip.

Sarà certamente possibile nazionalizzare il *Workbench*, anche con messaggi definiti dall'utente. Carolyn Scheppner ha fatto vedere in pratica quanto sia semplice ridefinire i testi dei messaggi di sistema. Altre novità: si potranno vedere come icone anche i file privi del suffisso ".info"; le finestre del dispositivo Console potranno avvalersi di un refresh basato su una matrice di caratteri (in altre parole, ridimensionando una finestra CLI non si perderà più una parte del testo).

Sono state migliorate anche le routine che gestiscono i layer, che ora come ora concedono troppo spesso "pause per il caffè" a chi sposta una finestra. Grazie alle Fat Keys sarà possibile riprogrammare la tastiera, come attualmente permettono di fare alcuni programmi specifici. Questa funzione non sarà però efficace con quei programmi (ancora troppi) che fanno una scansione diretta della tastiera. A ogni gadget di Intuition i programmatori potranno associare una funzione personalizzata per il controllo della grafica e degli ingressi dell'utente. In questo modo, sarà facile creare dei Radio Buttons come sul Macintosh. Migliorata anche la gestione dei gadget stringa. La versione 1.4 darà pieno supporto anche al modo video overscan e alle ColorFont (fonti-carattere di diversi colori). Sarà possibile lavorare contemporaneamente su più porte seriali e parallele. Quasi sicuramente ci sarà già nella versione 1.4 anche un file requester di sistema (ma sull'aspetto che dovrebbe avere sembrava che ognuno degli intervenuti avesse una sua opinione personale).

***Sopra:** la scala a chiocciola che porta ai saloni della Developers Conference. **Sotto:** l'intervento dell'oratore italiano Michele Console Battilana della Cloanto*

Attualmente, per aggiornare il sistema operativo alla versione 1.3 è sufficiente procurarsi il nuovo *Workbench*, dato che l'unica miglioria del *Kickstart 1.3* è la capacità di eseguire l'auto-boot da periferiche diverse da df0:. Nella versione 1.4 sono previste invece molte novità anche per il *Kickstart* e quindi in ROM (ad esclusione dell'Amiga 1000). Le versioni pre-release circolanti a Francoforte si servivano del cosiddetto *Jumpstart*, una tecnica provvisoria per utilizzare nuove funzioni a mano a mano che vengono scritte, senza dover ricorrere ogni volta a una nuova ROM. Per un miglior supporto dei nuovi modi grafici saranno inserite direttamente in ROM anche nuove fonti-carattere e più veloci routine per la gestione del testo. Per far posto alle innovazioni in ROM verrà probabilmente tolto il disegno della mano che invita a inserire un disco.

Il monitor ad alta risoluzione A2024 non avrà più bisogno di un *Workbench* apposito per funzionare. Come ci ha infatti spiegato Andy Finkel, anche i relativi codici saranno in ROM. Lo stesso vale per il nuovo Fast Filing System, che funzionerà anche sui dischi da 3,5", pur garantendo la compatibilità con

le versioni precedenti che agivano solo sull'hard disk. La versione 1.4 del sistema operativo funzionerà su qualsiasi Amiga, mentre il nuovo set di chip (Denise, Fat Agnus e Gary) potrà essere montato solo sugli Amiga 500 e 2000B. Messi alle strette dalle domande dei meno fortunati possessori degli altri sistemi, i responsabili della sezione marketing e vendite della Commodore hanno reso noto che «prenderanno in considerazione l'opportunità di attuare a condizioni di favore un upgrade a 2000B dei vecchi Amiga 2000A» (esternamente il 2000B si distingue dal 2000A perché ha in più l'uscita video monocromatica BAS).

Qualche novità software è emersa anche nella mostra delle ditte che partecipavano alla conferenza. La Lattice esponeva tutta la gamma dei suoi prodotti per l'Amiga. Tra le novità il compilatore C 5.0, il C++ e, finalmente, un ottimo debugger. Siamo riusciti a scoprire anche un'interessante caratteristica che verrà prossimamente aggiunta al compilatore. Sarà possibile creare librerie e dispositivi senza dover scrivere una sola riga in linguaggio macchina. Basterà informare il compilatore e il linker, con opportuni flag, di come verrà utilizzato il codice da compilare.

Un Amiga in versione "Tower" realizzato da un produttore di hardware tedesco

La sorpresa più piacevole, però, è stata vedere che la postazione più grande e affollata era quella dell'italianissima Cloanto. Tra le novità presentate dalla ditta di Udine il sistema di comunicazione per non vedenti e disabili HTX, che con le sue prestazioni innovative ha già imposto nuovi standard nel settore. C'era anche la versione finale del word processor Cloanto *C1-Text*, ormai utilizzato anche a West Chester per la sua capacità di leggere e convertire testi nei formati più svariati.

Soltanto a pochi distributori veniva invece concesso di vedere una pre-release del *Printer Fonts Maker*, un programma che permette di creare set di fonti-carattere da trasferire su qualsiasi stampante (download). I programmi presentati dalla Cloanto erano versioni multilingue, con interfaccia utente in inglese, tedesco e italiano.

Presso un'altra postazione abbiamo provato *ARexx*, una versione per l'Amiga del linguaggio *Rexx*, originariamente sviluppato dalla IBM. *Rexx* è un linguaggio interpretato moderno e potente, che può essere utilizzato anche come linguaggio Shell e per scambiare informazioni e comandi tra diversi programmi forniti d'interfaccia *Rexx*. Una delle caratteristiche del *Rexx* è infatti la possibilità d'interagire con funzioni esterne, cioè definite in altri programmi in esecuzione. Questa funzione rende *Rexx* idoneo a sostituire i linguaggi "macro" di qualsiasi applicazione.

Purtroppo c'era anche del software che nessuno si augurava di vedere: quasi tutti i calcolatori erano infettati da nuovi virus, sfuggiti al controllo degli antivirus più sofisticati. A ogni operazione di lettura/scrittura, gli hard disk di alcuni Amiga si riempivano tanto che si doveva ricorrere alla riformattazione.

Le novità non sono mancate neppure sul fronte hardware. Per una descrizione più dettagliata di alcuni dei prodotti qui citati si veda anche *Commodore Gazette* n. 2/1988, pag. 57 e seguenti. I computer a disposizione degli intervenuti montavano il nuovo chip Fat Agnus, che permette di lavorare con un megabyte (2 MB sugli Amiga del futuro) di Chip RAM. Ben quattro computer funzionavano con la scheda A2620, dotata di microprocessore 68020 e RAM a 32 bit. Questa scheda è stata definita dalla Commodore «un primo passo verso l'Amiga 3000».

Abbiamo finalmente potuto provare anche la scheda Bridgeboard AT A2286. Tra le caratteristiche di questo sistema spiccano l'emulazione del mouse Microsoft, un megabyte di RAM e disk drive in grado di lavorare altrettanto bene con dischi da 1,2 MB e da 360K. Le schede A2286 e A2620 possono funzionare insieme, senza alcun problema di compatibilità.

Jeff Porter ha spiegato che la nuova scheda d'espansione RAM A2058 è stata progettata tenendo presente l'esigenza di occupare un solo slot per espansioni di memoria, nella convinzione che il mercato delle memorie punti decisamente verso i chip da un megabyte. Utilizzando questi chip si potranno espandere i 2 MB della scheda fino a 4 o 8 MB di RAM, senza doversi preoccupare di aggiungere altre schede.

Per venire incontro ai possessori del controller A2090, che non possono usufruire della funzione di autoboot da hard disk del *Kickstart 1.3*, è stata realizzata la scheda A2090B. Tale scheda, che da sola occupa uno slot, contiene due EPROM che consentono di effettuare l'autoboot tramite la A2090 (e solo con questa), come se fossa la nuova A2090A.

Dovrebbe essere imminente la distribuzione dell'unità hard disk più espansione RAM A590 per l'Amiga 500. Il dispositivo, che si serve di un nuovo chip custom come interfaccia hardware-DMA, è completo d'interfaccia SCSI DB25 compatibile Macintosh. L'A590 viene collegato all'Amiga

500 attraverso il connettore laterale. L'alimentatore è esterno, ma si attiva automaticamente all'accensione del computer. Fino a 2 MB di RAM possono essere aggiunti in un secondo momento sulla scheda, che viene venduta con gli zoccoli vuoti. Una ventola evita il surriscaldamento. L'hard disk, da 20 MB con tempo di accesso di 80 msec (decisamente lento, come ha riconosciuto lo stesso Porter, aggiungendo: «Ma voi volete proprio tutto!») verrà consegnato già formattato e con il software della versione 1.3 del sistema operativo. L'Amiga 500 sarà in grado di effettuare automaticamente l'autoboot dall'A590.

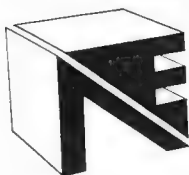
La scheda A2232 contiene sette porte seriali RS-232C, con velocità di trasmissione fino a 19.200 baud. Un particolare curioso: la scheda contiene un microprocessore 6502 (lo stesso del primo PET Commodore), che va programmato tramite il 68000 per alleggerire il carico di quest'ultimo. Qualche simpaticone ha osservato che evidentemente alla Commodore non sapevano più cosa farsene delle vecchie schede del VIC 20.

Molto interessante anche il Professional Video Adapter A2350. La scheda, progettata dal laboratorio grafico del New York Institute of Technology, riunisce le funzioni di genlock, digitizer e frame freeze. Il segnale video viene convertito e gestito internamente in modo digitale. La qualità dell'immagine "congelata" è paragonabile a quella dei videoregistratori Super VHS dotati di memoria. Non esistono switch hardware. Tutte le funzioni sono infatti regolabili per mezzo di gadget sovraimpressi all'immagine. Peccato che il tutto non supporti la risoluzione verticale del modo PAL overscan, cioè il modo richiesto da tutti gli operatori del settore video al di fuori degli Stati Uniti. Con grande imbarazzo, dopo un fuoco di fila di domande degli esperti video presenti in sala, i portavoce ufficiali della Commodore hanno infatti dovuto annunciare che la risoluzione massima di 1024 x 512 è strettamente collegata a una scelta hardware di base.

Gail Wellington ci ha confermato che l'Amiga 2000HD (Amiga 2000 più A2090A con hard disk da 40 MB, 28 msec) e l'Amiga 2500 (Amiga 2000HD più A2620) verranno distribuiti solo negli Stati Uniti. Negli altri Paesi si potrà ottenere la stessa configurazione mettendo insieme le diverse schede. Con il sistema Amiga 2500 UX (che abbiamo descritto nel numero 2/88 di *Commodore Gazette*) dotato di sistema operativo Unix System V versione 3.1, la Commodore intende affermarsi nell'ambiente universitario. Johann George, l'esperto Unix presente alla conferenza, ci ha rivelato che per la distribuzione del software di questa stazione di lavoro la Commodore ha deciso di servirsi delle cartucce a nastro in formato standard AT&T, utilizzate anche per effettuare il backup dell'hard disk. Un futuro sistema Unix della Commodore si servirà di un 68030 per rendere ancora più interessante il rapporto prezzo-prestazioni, e poter così penetrare maggiormente in quei mercati dove è necessaria la "forza bruta", a un prezzo evidentemente contenuto. La Commodore ha intanto lanciato una massiccia campagna d'informazione attraverso la stampa specializzata per scrollare di dosso dai marchi "Commodore" e "Amiga" la fama di macchine destinate soltanto ai videogiochi.

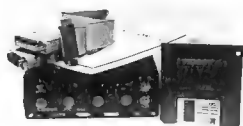
Lauren Brown ha annunciato che la prossima Developers Conference si terrà probabilmente a San Francisco dal 15 al 17 giugno 1989. È intenzione della Commodore organizzare un convegno come questo ogni sei mesi, alternando Europa e Stati Uniti. La conferenza si è conclusa con un sorteggio che ha visto i fortunati vincitori ricevere come premi l'Enhanced Chip Set e il compilatore C 5.0 della Lattice. ■

È IN EDICOLA IL NUMERO DI FEBBRAIO

Ultimobyte Editrice
Via A. Manuzio, 15
20124 Milano
Tel. 02/6597693

A CASA TUA
DIRETTAMENTE
02/33000036

AMIGASHOP

UNICA SEDE: VIA MAC MAHON, 75 - 20155 MILANO

Tel.: 02/323492 solo per negozio e informazioni relative acquisti in Milano - direttamente in sede
Tel. 02/33000036 per ordinazioni da tutta Italia; Fax 02/33000035 in funzione 24 ore su 24
BBS MODEM 02/3270226 (banca dati) al pomeriggio dopo le 13.00 fino al mattino successivo
Aperto al pubblico nei giorni feriali dalle 9.00 alle 12.30 e dalle 15.00 alle 19.00
e il sabato dalle 9.30 alle 13.00 e dalle 14.30 alle 18.30 - chiuso il lunedì

VIDEON

Basta con i noiosi filtri per i vari passaggi... Ora c'è VIDEON! Il VIDEON è un digitalizzatore video a colori dotato di un convertitore PAL-RGB con una banda passante di 15 KHz per ottenere immagini a colori dalle stupefacenti qualità... Funziona in risoluzioni di: 320x256 - 320x512 - 640x256 - 640x512. Può essere collegato a una qualsiasi fonte video PAL, ad esempio videoregistratori, computer, telecamere, televisori, ecc. Il prodotto permette di visualizzare il segnale video collegato all'apparecchio e in più permette la regolazione di luminosità, colore, saturazione, contrasto.
È corredato di software che permette la manipolazione di immagini IFF HOLD MODIFY da 32 a 4096 colori con tecniche di SURFACE-MAPPING su solidi geometrici.
È in arrivo la versione 2.0

L. telefonare

FLICKER FIXER
Novità in arrivo.
Questa eccezionale scheda che si inserisce nell'A2000 toglie il fastidioso Flicker dell'Amiga che si verifica in altissima risoluzione. Per chi usa l'Amiga per lavoro o con grafica CAD, ecc.
L. telefonare

AMIGA SPLITTER
Per chi già possiede un digitalizzatore video del tipo Amiga Eye, Amiga VID, Easy View, Digi View 3.0, ecc.
Evita il passaggio dei noiosi tre filtri. Lo splitter converte direttamente l'immagine a colori, indispensabile per chi possiede un digitalizzatore normale.
L. 199.000

MINI GEN

MINI-GEN una grande novità per professionisti ed entusiasti, per ottenere sovrapposizioni di animazioni, titoli, messaggi ecc.
Funziona con tutti gli Amiga ed è compatibile con programmi come TV-Text, Pro video e molti altri. Ora la videotitolazione è alla portata di tutti, semplicissimo da usare.
L. 399.000

KICKSTART 1.3 ROM
Il nuovo sistema operativo dell'Amiga ora in ROM applicabile facilmente su A500 e A2000 senza saldature e senza perdere il vecchio s/o 1.2.
L. 119.000

TASTIERA
musicale, Amiga compatibile. Pro Sound designer
New

ESPANSIONI
512K originali Commodore per A500
L. 319.000
2MB esterne autoconfiguranti profex per A500
L. 1.090.000
Disponibili espansioni di memoria per A500, 1000, 2000 interne ed esterne da 512K fino ad 8MB. Telefonare per ulteriori informazioni.
L. telefonare

AMIGA FAX
Straordinaria novità per ricevere segnali, fax, cartine, meteo, ecc. con il tuo Amiga, composto da: scheda hardware, software di gestione, manuale d'uso.
L. 199.000

PRO SOUND DESIGNER

Ovvero Elaboratore professionale del suono. È un campionatore sonoro che funziona su tutti gli Amiga, 8 bit stereo sampler da 1 a 28 KHz mono e da 1 a 17 KHz stereo; playback a 35 KHz, avanzate funzioni di editing e compatibile anche con altri pacchetti software come ad esempio : Sound sampler, Future sound, Perfect sound, ecc.
L. 179.000

AMIGA SCANNER
Nuovo scanner grafico per Amiga, copia un testo, una foto, un disegno sul computer ed è in grado di modificarlo velocemente con i suoi numerosi programmi.
L. 799.000

AMIGA CARD

Hard disk in AmigaDOS per l'Amiga 2000 su scheda, semplice da installare e lascia libero lo spazio per il secondo drive interno. Disponibili anche versioni esterne per A500 e A1000.
L. 990.000

AMIGA MODEM 2400 PAK
Modem dedicato per A500 - A1000 - A2000, esterno 300, 1200, 2400 baud (V21-22-22BIS). Autodial, autoanswer, Hayes compatibile, completo di software e cavo di connessione al computer (disponibili altre versioni, 300/1200 e 300/1200-1200/75 Videotel).
L. 399.000

TELECAMERA B/N
Alta risoluzione (600 linee) da accoppiare a DIGIVIEW, EASY VIEW, REALTIME, VID VIDEON ecc.
L. 399.000

Stampanti 9 e 24 aghi colore o bianco e nero: NEC, STAR, PANASONIC, AMSTRAD, TAXAN, COMMODORE, OLIVETTI, MANNESMANN ecc. a prezzi da grossista.

Genlock Broadcasting Neriki (per STUDI e/o TV private, alte prestazioni).
L. telefonare

Stazioni Grafiche composte da Amiga 2000 Hard Disk (20-32 MB) Scheda Janus, 2MB-8MB, Genlock e programmi grafici.
prezzi concorrenziali

Scheda XT Janus (compatibilità 100% MS DOS) **e AT Janus per Amiga 2000**
L. telefonare
Sono disponibili i programmi di Fred Fish, di Public Domain e relativo manuale d'uso in italiano.
L. 2000 il catalogo

I NOSTRI DISK DRIVE
DISK DRIVE SLIM, MECCANICA NEC BEIGE

sono disponibili:
per Amiga 500 3,5 pollici passante compreso disconnect **L. 239.000**
per Amiga 500 5,25 pollici 40/80 tracce passante **L. 350.000**
per Amiga 2000 interno **L. 179.000**
per C-64 OCC118 **L. 239.000**

ACCESSORI PER L'AMIGA

EASY SOUND	L. 119.000	Digitalizzatore Audio IFF compatibile Sonix ecc.
EASY VIEW	L. 119.000	Digitalizzatore Video compatibile Digiview
DIGI AUDI & VIDEO	L. 189.000	Digitalizzatore Audio & Video: tutto in uno come sopra
SERIAL AMIGA	L. 39.000	Interfaccia per collegare stampanti seriali/64 all'Amiga
INT. MIDI	L. 59.000	Per collegare tastiere MIDI all'Amiga (con software)
INT. MIDI PROF.	L. 79.000	Per collegare tastiere MIDI all'Amiga (Passthrought)
PAL GENLOCK	L. 590.000	Genlock amatoriale con regolazioni per A500, A1000, A2000
PORTADISCHI 40pz.	L. 20.000	
PORTADISCHI 60pz.	L. 30.000	
KIT PULIZIA 3,5"	L. 10.000	
KIT PULIZIA 5,25"	L. 10.000	
PORTADISCHI POSSO	L. 34.900	(100 posti)

Per questioni di spazio non ci è possibile elencare moltissimi altri articoli. Veniteci a trovare o richiedeteci i cataloghi settoriali. Chi verrà a trovarci con questa rivista e acquisterà almeno L. 100.000 (centomila) di prodotti accessoristici hardware e software riceverà, richiedendolo, un "controvalore" pari al costo della rivista.

Alcuni dei nostri prodotti, costruiti, importati o distribuiti da Newel si possono trovare anche da PERSONAL COMPUTER PESARO

Tutto il materiale è garantito 12 mesi + 7 giorni di prova soddisfatti o rimborsati ed è in pronta consegna = NOI VENDIAMO FATTI, NON PAROLE = I nomi, i marchi e gli stemmi usati in questa pubblicità sono depositati e di proprietà delle menzionate aziende, Newel ne è solo il rivenditore, o il distributore, e ringrazia le medesime per l'utilizzo.
I PREZZI POSSONO VARIARE SENZA PREAVVISO

Spedizioni in contrassegno postale in tutta Italia in REALTIME (servizio computerizzato) 02/33000036 (da martedì a venerdì dalle ore 9,15 alle 18,50)
Servirsi per ordini esclusivamente dei numeri indicati o del Fax

SCONTI a scuole, comunità, associazioni e professionisti.
I signori rivenditori sono pregati di visitarci il lunedì, previo appuntamento.

Si cercano collaboratori part-time o tempo pieno per i settori da noi trattati; espertissimi e già introdotti nel settore (telefonare al mattino presto 02/323492)

Ricorda alla Newel trovi anche tutto per C-64/128, Amiga, Atari, PC Amstrad ecc.
Richiedi il catalogo specificando il computer posseduto.
Richiedi il nostro nuovo catalogo per Amiga con tutte le ultime novità hardware & software, oltre 1000 programmi selezionati... inviando L. 2.000 in francobolli.

Nuovo servizio: se hai dubbi su qualche prodotto, te lo diamo in prova per 48 ore a casa tua (dietro cauzione) e se non ti soddisfa ti restituiamo i soldi senza formalità, purché sia restituito nello stato iniziale.

DISCONNECT
Super interfaccia, che permette di scollegare i disk drive esterni dell'Amiga senza spegnere il computer, escludendoli all'istante e ricollegandoli quando serve. Con questo sistema potete usare tutti i programmi che necessitano di una quantità di memoria superiore a quella residua con l'uso di due o più unità disco.
L. 23.000

HARD DISK CARD
per A2000 in modo MS-DOS (meccanica Miniscribe, Controller Westerndigital)

20 MB	**L. 639.000**
33 MB	**L. 799.000**
40 MB	**L. 969.000**
20 MB HARD DISK	**L. 539.000**

CHIAVI IN MANO - PREZZI IVA INCLUSA - TUTTO COMPRESO

L'ARCHITETTURA DEL SISTEMA OPERATIVO GEOS: QUINTA PARTE

GEOS: COME COSTRUIRE UN BOX DI DIALOGO

Concludiamo questa serie illustrando i box di dialogo, strumenti indispensabili per gestire semplici e immediate comunicazioni con l'utente. Due esempi illustrano le strutture di definizione di due diversi box di dialogo

di Luca Giachino

(N.d.r.) Questo articolo s'inserisce in un ciclo dedicato al sistema operativo GEOS. Per una lettura agevole è necessario conoscere gli argomenti trattati nelle precedenti puntate.

Nelle precedenti puntate di questa serie abbiamo esplorato molti tra gli aspetti più significativi della struttura del sistema operativo *GEOS*. Abbiamo detto che si tratta di un sistema operativo a gestione di eventi, e che le applicazioni a esso dedicate sono definite a flusso di eventi. Un "evento" è qualsiasi azione che richiami l'attenzione del sistema. Se l'evento è prodotto dall'utente, deve trattarsi di un tentativo di comunicazione con il sistema, come il movimento del mouse, la selezione di un menu o di un'icona, oppure un qualsiasi input da tastiera. Gli eventi che si verificano allo scadere di un conto alla rovescia sono invece i cosiddetti "eventi temporizzati", e il conto alla rovescia è gestito da un contatore relativo a un processo in corso.

Abbiamo spiegato che la routine MainLoop si prende cura di tutti gli eventi definiti dall'applicazione, chiamando all'occasione le relative routine di servizio.

Dopo questa panoramica sugli eventi, e dopo averli classificati nei tre tipi fondamentali, abbiamo esaminato l'interazione fra le due routine più importanti in tutto il sistema: MainLoop e InterruptMain. Si è detto che la prima è il cervello e la seconda il sistema nervoso del sistema operativo *GEOS*, cercando con questa similitudine di renderne chiari i rispettivi compiti.

Abbiamo poi rivolto la nostra attenzione verso gli strumenti interattivi a disposizione delle applicazioni, iniziando dai menu e dalle icone. Si è cercato di mettere in luce soprattutto il carico di lavoro svolto dal sistema per agevolare le applicazioni: con la creazione in memoria di semplici strutture di dati, il sistema visualizza il menu e le icone e procede alla loro gestione autonomamente, senza l'intervento dell'applicazione. Nelle strutture di dati, infatti, l'applicazione ha la possibilità d'indicare posizioni sullo schermo, dimensioni geometriche, stringhe a terminazione nulla per le voci di tutti i sotto-menu, disegni delle icone, tipi di reazione alla selezione e indirizzi delle routine di servizio relative agli eventi finali.

L'applicazione, chiamando semplicemente le routine DoMenu e DoIcons, cede queste strutture di dati al sistema, che procede alla gestione degli strumenti interattivi.

Ci siamo poi occupati delle capacità multitasking del sistema operativo *GEOS*, mettendone in evidenza pregi e limiti. In particolare abbiamo analizzato i processi temporizzati e i desk accessory. Questi ultimi ricalcano esattamente il sistema adottato dal Macintosh per offrire una parvenza di capacità multitasking: il sistema esegue una sola applicazione alla volta, ma può congelare quella in esecuzione per cedere temporaneamente il controllo a un'altra (il desk accessory). Terminata l'esecuzione del desk accessory, il controllo torna all'applicazione principale.

In questa puntata finale ci dedicheremo ai box di dialogo, strumenti interattivi molto simili ai desk accessory, perché anch'essi sono piccole applicazioni temporanee, con nuovi menu e icone.

I box di dialogo

Nelle applicazioni *GEOS* compatibili i box di dialogo sono molto frequenti. Ogni volta che dobbiamo inserire il nome di un file, selezionare un file di testo con *geoWrite*, o mandare in stam-

pa un'immagine con *geoPublish*, sullo schermo appare un box tramite il quale l'applicazione sottopone all'utente un ventaglio di possibili scelte. Il box di dialogo è quindi uno strumento di dialogo con l'utente come menu e icone, anche se ha caratteristiche completamente diverse.

Mentre i menu e le icone sono sempre presenti sullo schermo, i BD (abbreviazione di box di dialogo) appaiono soltanto quando si presentano particolari necessità, e impegnano quindi lo schermo solo temporaneamente. Inoltre, i box di dialogo sono in realtà delle vere e proprie applicazioni, come i desk accessory, nel senso che possono essere strutturati con menu e icone proprie, e propri codici d'inizializzazione degli eventi. Quest'ultimo caso, indubbiamente il più complesso, prevede per le applicazioni un maggior carico di lavoro, in quanto le strutture standard previste dal sistema devono essere necessariamente scavalcate.

La gestione dei box di dialogo

Quando un'applicazione deve visualizzare un BD, chiama la routine DoDlgBox indicando in un apposito registro l'indirizzo della struttura di dati che lo definisce. Questa struttura è costituita da una serie di comandi, e relativi parametri, che il sistema legge ed esegue in sequenza quando viene chiamata DoDlgBox. Si tratta di 19 comandi, di cui i primi dieci servono per visualizzare le icone di sistema. Ogni comando è contraddistinto da un codice formato da un byte, ed eventualmente da alcuni parametri specificati nei byte che seguono. Se il sistema incontra il codice di un comando a zero, lo interpreta come fine della struttura di dati.

La struttura di un BD dev'essere considerata come la routine d'inizializzazione di un'applicazione. Sebbene non contenga codici eseguibili ma solo byte comandi, le sue funzioni sono praticamente identiche: configurare il box, e inizializzare i nuovi eventi di cui il sistema dovrà temporaneamente prendersi cura. Non c'è conflitto con l'applicazione principale dal momento che il sistema garantisce l'incolumità di tutti i suoi dati. Per la precisione, in una particolare area RAM di proprietà del sistema vengono salvate le variabili globali del Kernel, i vettori e le strutture dei menu e delle icone. Quando il box di dialogo viene chiuso, queste informazioni (lo stato del sistema) vengono completamente ripristinate, e il controllo viene restituito all'applicazione dopo l'istruzione jsr DoDlgBox, come se niente fosse accaduto. Il BD viene infatti cancellato dallo schermo, rivelando quanto aveva nascosto.

Esistono due tipi di gestione del BD, quella standard e quella non standard. La gestione standard prevede l'apparizione sullo schermo di un box e un'ombra di dimensioni e posizione predefinite. Questo significa che nella struttura di definizione non si devono inserire posizioni e dimensioni del box. L'utilità di un box standard è evidente: se non sono necessarie dimensioni particolari, si risparmia qualche byte nella struttura di definizione. Inoltre la gestione standard prevede l'impiego delle icone di sistema e delle consuete operazioni di I/O (visualizzazione di messaggi e input di stringhe). In questo tipo di gestione, la selezione di un'icona o la pressione del tasto Return durante l'immissione di un testo provoca l'automatica chiusura del BD. L'applicazione rileva la causa che ha prodotto la chiusura del BD analizzando il contenuto di un particolare parametro di ritorno. Esempio di BD definito con strutture standard è il box di dialogo che appare quando si desidera cambiare il nome di un file da *deskTop*, oppure formattare un disco. Ma talvolta questo tipo di gestione non è sufficiente, e occorre ritoccare la struttura di definizione del BD con nuovi comandi, passando alla gestione non standard.

La gestione non standard si riferisce appunto alle strutture di definizione dei BD che non ricorrono ai box di dimensioni predefinite con icone predefinite. La gestione non standard però, oltre che dimensioni e icone proprie prevede anche alcuni comandi che possono aumentare la complessità del BD a qualsiasi livello, trasformandolo in una piccola applicazione. In particolare, diventa possibile cedere al sistema una stringa grafica (una stringa di comandi grafici che il sistema esegue in sequenza), richiedere la gestione standard per la selezione dei file presenti sul disco e infine mandare in esecuzione una routine che costituisce il proseguimento della procedura d'inizializzazione.

Comunque, a prescindere dal tipo di gestione, quando il BD è stato aperto, il controllo passa di nuovo a MainLoop, la quale si preoccupa di gestire tutti i nuovi eventi, standard e non, previsti in fase di definizione del BD. Questo spiega perché un BD può essere considerato una piccola applicazione. Inoltre, se le routine di servizio coinvolte terminano con l'istruzione rts, ricedono il controllo a MainLoop come accade per ogni altra routine di servizio. Nel caso dei BD questo significa che la gestione degli eventi riprende e che il BD non viene chiuso. Per ovviare a questa situazione e per fare in modo che le routine di servizio possano chiudere il BD, il Kernel di *GEOS* prevede la funzione RstrFrmDialogue, che provvede a eseguire la procedura standard di chiusura del BD (ripristino di tutte le variabili di sistema e della parte di schermo nascosta dal box). Questa funzione si aspetta che la routine di servizio chiamante abbia memorizzato nella variabile di sistema sysDBData il codice corrispondente alla causa della chiusura (in genere il codice del comando che ha prodotto la chiusura, ma se la gestione non è standard potrebbe trattarsi di un qualsiasi altro valore). RstrFrmDialogue, infatti, copia il contenuto di que-

sta variabile globale nel registro r0L che restituisce ai codici del programma.

I comandi dei box di dialogo

Per comprendere meglio il funzionamento dei BD, è opportuno analizzare i comandi di cui disponiamo per creare la struttura di definizione. Questa struttura deve sempre iniziare con un byte che indica al sistema se, visualizzando il box, deve usare o meno le dimensioni predefinite, e che indica inoltre il tipo di matrice grafica da impiegare per l'ombra (nel Kernel di *GEOS* sono definite ben 32 matrici grafiche; per vederle basta osservare gli sfondi che *geoPaint* mette a disposizione). Se questo byte indica che il box non è standard, i byte successivi devono individuare le coordinate dell'angolo superiore sinistro e dell'angolo inferiore destro. Queste coordinate sono relative all'angolo superiore sinistro dello schermo, e sono le uniche coordinate assolute nella struttura di definizione di un BD: tutte le altre (posizioni delle icone, delle stringhe da visualizzare...) sono relative all'angolo superiore sinistro del BD. Dopo il byte che indica al sistema come disegnare il BD, gli altri comandi possono succedersi in qualunque numero e in qualunque ordine. Analizziamoli uno per uno.

– I comandi dall'uno al dieci visualizzano e attivano nel BD particolari icone dette icone di sistema. Fino alla versione 1.3 del Kernel, di queste dieci icone sono disponibili solo le prime sei. Si tratta delle familiari icone Ok, Cancel, Yes, No, Open, Disk, utili per una grande varietà d'impieghi. I vantaggi che presentano sono fondamentalmente due: i loro disegni sono predefiniti nel Kernel, e per usarle basta indicare l'opportuno codice e la posizione che devono avere all'interno del BD. Come si potrà notare, un carico di lavoro molto minore rispetto all'abilitazione delle generiche icone. Inoltre, non vi è necessità di specificare alcuna routine di servizio, perché le routine di servizio relative a queste icone sono predefinite nel sistema. In realtà il sistema prevede per tutte queste icone un'unica routine di servizio che memorizza il codice dell'icona selezionata nella variabile di sistema sysDBData, e chiama la routine RstrFrmDialogue. Quest'ultima memorizza il valore di sysDBData nel registro r0L, e provvede a eseguire la procedura standard di chiusura del BD. Quindi, le icone predefinite causano sempre la chiusura del BD.

– **DBTXTSTR** è il comando che visualizza una stringa a terminazione nulla nel BD. È seguito dalle coordinate del primo carattere che deve apparire nel BD e dal puntatore che individua la stringa in memoria.

– **DBVARSTR**, al pari del precedente, permette di visualizzare una stringa nel BD. L'unica differenza è che il comando è seguito dall'indirizzo del registro (da r0 a r10, r15) che contiene il puntatore alla stringa da visualizzare. Questa è un'informazione dinamica, che può cambiare ogni volta che il BD viene visualizzato. Con questo comando si può quindi visualizzare una stringa diversa alla stessa posizione ogni volta che si riapre il BD.

– **DBGETSTRING** richiede al Kernel di ricevere in input una stringa a una particolare posizione e per un particolare numero di caratteri. Il comando è seguito dalla posizione in cui deve apparire il cursore, dall'indirizzo del registro che individua il buffer in memoria, e dal massimo numero di caratteri che il buffer può contenere. Si noti che se nel buffer è già presente una stringa a terminazione nulla questa verrà visualizzata nel box come prima scelta possibile. L'utente può semplicemente premere il Return o decidere di alterarla. Ovviamente, l'input della stringa viene trattato dal sistema come un evento, cioè viene gestito parallelamente a tutti gli altri eventi che caratterizzano il BD. Quando l'utente preme il Return, il BD viene chiuso; i codici chiamanti possono quindi esaminare il parametro restituito ed eventualmente il contenuto del buffer.

– **DBSYSOPV** indica al sistema di gestire in modo standard la selezione del mouse in un'area non convenzionale (un'area non controllata dal sistema), cioè con la semplice chiusura del BD. Questo comando risulta utile per esempio quando si desidera mostrare con il BD un semplice messaggio, che l'utente chiude premendo il pulsante del mouse in qualunque punto dello schermo.

– **DBGRPHSTR** ordina al sistema di eseguire la stringa grafica individuata dall'indirizzo che segue il comando. Si ricordi che una stringa grafica è un array di comandi che, eseguiti in sequenza dal sistema, visualizzano elementi grafici come rettangoli pieni, cornici, linee oblique... Una stringa grafica è particolarmente comoda quando occorre visualizzare una serie di elementi grafici sempre uguale, dal momento che evita di dover chiamare ogni volta le stesse routine grafiche del Kernel. In genere le stringhe grafiche vengono usate dalle applicazioni chiamando la routine GraphicsString, ma il Kernel ne permette l'uso anche all'interno dei BD.

– **DBGETFILES** è un comando che visualizza nel BD, all'interno di una finestra di scroll, tutti i nomi dei file di un certo tipo presenti sul disco. Un esempio di programma che utilizza DBGETFILES è fornito da *geoWrite*, quando l'utente deve aprire un testo già presente sul disco. La particolarità di questo comando è che visualizza solo i nomi dei file appartenenti a un particolare tipo. Si ricordi infatti che *GEOS* è in grado di suddividere i file in 15 differenti tipi. Per esempio, quando si deve scegliere un nuovo driver di stampa, *deskTop* visualizza soltanto i nomi dei file di tipo PRINTER. Oltre a questa prima chiave di ricerca, peraltro alquanto generica, il comando permette di specificare anche una seconda chiave subordinata. Si tratta del

nome permanente di cui ogni applicazione deve fornire i file dati che crea. L'esigenza nasce dal fatto che le applicazioni, *geoWrite*, *geoPaint*, *geoChart* o qualsiasi altra, creano sempre file di tipo APPL_DATA, e questo identificatore non è sufficiente per raggruppare i file prodotti da una particolare applicazione. Il nome permanente risolve il problema restringendo ulteriormente il campo di ricerca. Con questo comando, come con la funzione FindFTypes, *GEOS* è in grado di raggruppare un massimo di 16 file. Il nome del file selezionato dall'utente viene inserito nell'apposito buffer indicato dopo il comando.

– **DBOPVEC** permette d'indicare al sistema una particolare routine di gestione dell'evento "selezione di un'area non convenzionale". A differenza del comando DBSYSOPV, DBOPVEC permette alle applicazioni di definire routine di gestione personalizzate. La routine di servizio che viene individuata da questo comando può terminare con un semplice rts, nel qual caso il BD non viene chiuso e il controllo torna a MainLoop, oppure con l'istruzione jmp RstrFrmDialogue, la quale chiude definitivamente il BD.

– **DBUSRICON** permette di visualizzare e attivare nel BD un'icona non standard. Oltre alle coordinate dell'icona, questo comando dev'essere seguito dall'indirizzo della struttura di dati che definisce l'icona. Per le icone che vengono definite in questo modo, il Kernel non prevede routine di servizio predefinite, e si aspetta che nelle loro strutture di definizione appaiano gli indirizzi delle relative routine di servizio. L'applicazione deve quindi decidere sotto ogni aspetto il comportamento dell'icona. Per la fine delle routine di servizio di queste icone valgono le solite considerazioni: rts per non chiudere il BD, o jmp RstrFrmDialogue per chiuderlo.

– **DB_USR_ROUT** è l'ultimo comando previsto dal Kernel, e anche il più flessibile, in quanto permette di personalizzare completamente il BD. Infatti, dev'essere seguito dall'indirizzo di una routine d'inizializzazione alla quale il Kernel cede il controllo durante l'apertura del BD. Questa routine, come quella che riceve il controllo quando un'applicazione viene caricata in memoria, può impostare qualsiasi tipo di evento (temporizzazioni, menu, icone, selezioni dell'utente su aree non convenzionali...). In pratica, scavalcando i limiti imposti dai comandi visti finora, rende il BD una vera e propria applicazione temporanea. Ovviamente, deve terminare con l'istruzione rts in modo che il sistema possa procedere a eseguire i successivi comandi presenti nella struttura di definizione del BD. L'unica vera precauzione che occorre prendere con questa routine riguarda l'abilitazione di un set d'icone tramite la funzione DoIcons. Infatti, in questo caso è necessario che nell'intera sequenza dei comandi non ne appaia nessuno di quelli che attivano icone (si ricordi che in *GEOS* può essere attivo solo un set di icone alla volta, e che un nuovo set disabilita quello precedente).

Vale la pena d'insistere ancora sulla flessibilità del comando DB_USR_ROUT. In un ambiente di programmazione meno articolato, nella gestione dei BD sarebbe stato sufficiente soltanto questo comando, che in pratica delega all'applicazione la configurazione dell'intero BD. Fortunatamente, il Kernel di *GEOS* consente d'impiegarlo solo quando è strettamente necessario, e fornisce in alternativa tutti i comandi che abbiamo illustrato prima.

I limiti dei box di dialogo

Nella tavola della pagina successiva sono riportate come esempio le strutture di definizione di due BD. La prima è tratta direttamente dal Kernel di *GEOS*: definisce il BD che *GEOS* visualizza quando, terminata un'applicazione, cerca di caricare *deskTop* ma non la trova. La seconda struttura è invece di nostra creazione: definisce un BD all'interno del quale l'utente viene invitato a inserire un codice, può selezionare l'icona Quit, o servirsi di un menu. Si tratta di due semplici esempi che ovviamente non esauriscono neanche lontanamente le possibilità offerte dai BD, ma che sono utili per vedere in concreto come può essere organizzata la struttura di definizione di un box di dialogo.

A questo punto rimane soltanto da chiedersi se *GEOS*, pur trattando i BD proprio come se fossero applicazioni (al pari dei desk accessory), imponga loro qualche limitazione. La risposta è ovviamente affermativa, in quanto i BD, sebbene abbiano a loro disposizione l'intero sistema, convivono in memoria con le applicazioni che li aprono. Non possono aprire altri box di dialogo (al pari dei desk accessory che non possono aprire altri desk accessory) e richiedono alcune precauzioni per non intaccare il buffer di schermo.

Il buffer di schermo, ampiamente illustrato nelle scorse puntate, viene sempre alterato dai desk accessory e dai BD che prevedono una struttura di menu. Nella scorsa puntata avevamo visto come viene risolto questo problema nel caso dei desk accessory. Per quanto riguarda i BD, l'applicazione è ovviamente in grado di sapere se questo impiega o meno i menu. Se li impiega, prima di chiamare la funzione DoDlgBox, l'applicazione deve memorizzare nel vettore recoverVector l'indirizzo di una propria routine che sopperisca all'inattendibilità dei dati presenti nel buffer di schermo. Ogni applicazione impiega questo vettore a seconda di com'è stata strutturata, e i modi per risolvere il problema sono molti. Il più elegante è salvare in un buffer temporaneo la parte di schermo che verrà intaccata dal BD e dalla sua ombra ancor prima di aprirlo. Comunque, rispetto al caso dei desk accessory, è evidente che il ripristino dello schermo qui presenta meno problemi proprio perché il

comportamento del BD è perfettamente noto all'applicazione.

Conclusioni

Con questa puntata, concludiamo la breve serie di articoli dedicati all'esplorazione del sistema operativo *GEOS*. Abbiamo fornito moltissimi spunti per coloro che programmano in questo ambiente, ma vorremmo ricordare ancora che gli argomenti trattati costituiscono una minima parte di quanto si trova ampiamente illustrato nella *Guida ufficiale alla programmazione di GEOS*, edita dalla IHT Gruppo Editoriale, la casa editrice di *Commodore Gazette*. È solo con questo importante strumento di lavoro che diventa possibile conoscere a fondo il sistema, sfruttarlo al meglio, e produrre splendide applicazioni *GEOS* compatibili. Comunque, programmazione a parte, la serie si è rivolta soprattutto a coloro che vedendo *GEOS* si sono incuriositi, e hanno desiderato saperne di più, impadronirsi delle famose "chiavi del regno". Perché *GEOS*, come abbiamo visto in questi articoli, non è affatto un sistema operativo semplice: la sua complessità e flessibilità non trovano eguali nel mondo a 8 bit. Senza il suo avvento, soluzioni interattive d'avanguardia come i menu, le icone, i desk accessory, i box di dialogo, il mouse, e tante altre caratteristiche inedite, avremmo potuto goderle solo sul superbo schermo di un Macintosh, con la speranza inespressa di cambiare al più presto il nostro computer. *GEOS* ha reso disponibile per il C-64 quello che in scala è l'ambiente di lavoro del Macintosh. Anche se rimangono alcuni limiti che rendono problematico un impiego professionale di *GEOS*, non si può negare che in tutto il mondo, grazie al lavoro svolto dalla Berkeley Softworks, molti milioni di utenti del C-64 possono sperimentare il fascino di un ambiente di lavoro particolarmente user-friendly. Sotto il profilo didattico questo è un risultato di grande prestigio. ■

```
; *** Struttura di definizione del BD che appare quando il Kernel ***
; *** non riesce a trovare sul disco il programma deskTop ***

DeskTop:
 .byte   DEF_DB_POS|1        ;Dimensioni e posizione standard,
                             ;matrice grafica 1
 .byte   DBTXTSTR            ;Visualizza una stringa
 .byte   16                  ;a partire dalla posizione
 .byte   22                  ;relativa 16,22
 .word   Testo_1             ;Puntatore alla stringa

 .byte   DBTXTSTR            ;Visualizza una stringa
 .byte   16                  ;a partire dalla posizione
 .byte   38                  ;relativa 16,38
 .word   Testo_2             ;Puntatore alla stringa

 .byte   OK                  ;Visualizza l'icona standard Ok
 .byte   17                  ;alla posizione relativa
 .byte   72                  ;17,72

 .byte   0                   ;Fine della struttura di definizione
Testo_1:
 .byte   BOLDON,"Please insert a disk",0
Testo_2:
 .byte   "with deskTop V1.3 or higher",0

; *** Struttura di un BD che richiede un input dall'utente ***
; *** e attiva un piccolo menu all'interno del BD ***

CGBox_di_dialogo:
 .byte   SET_DB_POS|2        ;Dimensioni non standard,
                             ;matrice grafica 2
 .byte   latoSuperiore       ;Dimensioni del box
 .byte   latoInferiore
 .word   latoSinistro
 .word   latoDestro

 .byte   DBTXTSTR            ;Visualizza una stringa
 .byte   16
 .byte   22
 .word   InputStr            ;Puntatore alla stringa

 .byte   DBGETSTRING         ;Input di una stringa
 .byte   16
 .byte   38
 .byte   2                   ;Il buffer nel registro r0
 .byte   20                  ;deve accettare non piu' di
                             ;20 caratteri
 .byte   DBUSRICON           ;Visualizza un'icona
 .byte   2                   ;non standard
 .byte   24
 .word   IconaCreate         ;Struttura di definizione
                             ;dell'icona
 .byte   DB_USR_ROUT         ;Procedura non standard
 .word   MenuRoutine         ;e suo indirizzo

 .byte   0                   ;Fine della struttura di definizione
InputStr:
 .byte   BOLDON,"Inserisci il codice",PLAINTEXT,0

IconaCreate:
 .word   CreateDisegno       ;Struttura di definizione
 .byte   0                   ;dell'icona non standard
 .byte   0
 .byte   SYSDBI_WIDTH        ;Dimensioni dell'icona
 .byte   SYSDBI_HEIGHT
 .word   CreateRouServ       ;Routine di servizio
                             ;dell'icona

MenuRoutine:                 ;Procedura non standard
 LoadW   r0, MenuStruttura
 jsr     DoMenu              ;Attiva i menu per il BD
 rts                         ;Restituisce il controllo
                             ;a MainLoop

CreateRoutServ:              ;Routine di servizio dell'icona
 lda     #20                 ;Codice di ritorno
 sta     sysDBData           ;corrispondente all'icona
 jmp     RstrFrmDialogue     ;Chiude il BD

CreateDisegno:               ;96 byte di dati grafici dell'icona
```

GEOS 2.0

L'AVVENTURA CONTINUA

Si sente spesso dire «Chi s'accontenta gode», ma nel mondo di *GEOS* questo proverbio non ha diritto di cittadinanza. Gli utenti di *GEOS* non vogliono proprio saperne di accontentarsi, ansiosi come sono di sempre nuovi miglioramenti. All'uscita di *GEOS*, infatti, molti utenti hanno subito mosso intelligenti critiche a questo o quel particolare, individuando con precisione i punti deboli del nuovissimo sistema operativo. Ma si trattava di quel tipo di critica costruttiva che si rivolge a chi, avendo già fatto un buon lavoro, può fare anche di meglio. Lo stesso spirito sopravvive negli utenti di oggi, anche se miglioramenti e novità sono giunti in abbondanza. Si tratta di un segnale positivo, un continuo stimolo per i programmatori della Berkeley Softworks.

L'ultima e attesissima novità è *GEOS 2.0*: il pacchetto riunisce, oltre al Kernel e a *deskTop*, i tre pacchetti *geoWrite Workshop*, *DeskPack Plus* e *geoSpell*, offrendo un ambiente di lavoro abbastanza completo. Molti desk accessory sono stati migliorati tangibilmente, o persino riscritti, e lo stesso si può dire per alcune applicazioni, come *geoSpell*. Le novità introdotte cancellano con un colpo di spugna la maggior parte dei problemi rilevati finora dagli utenti e fanno sperare in un futuro sempre migliore. In uno dei prossimi numeri di *Commodore Gazette* svolgeremo una completa analisi del pacchetto, ma alcune anticipazioni possiamo darle già adesso.

GEOS 2.0

Il Kernel, dal punto di vista delle applicazioni, è rimasto immutato. Questo significa che la tavola dei vettori di salto alle funzioni del sistema e le dislocazioni delle variabili di sistema sono immutate. In effetti, ogni volta che la Berkeley introduce sul mercato una nuova versione di *GEOS*, un'analisi attenta dei codici rivela che in realtà il sistema è stato riordinato, ma la compatibilità con le applicazioni è sempre garantita. Le maggiori novità del pacchetto riguardano quindi le applicazioni che accompagnano il sistema operativo.

Quelle maggiormente rivoluzionate sono *deskTop*, giunta alla versione 1.5, *geoPaint*, *geoSpell*, *Configure*, e i desk accessory Photo Manager, Text Manager, Calculator, Notepad. Inoltre, sul disco è presente anche un nuovo e interessante desk accessory, Pad Color Manager, che permette di colorare le icone dei file. Ci risulta invece che le applicazioni contenute nei pacchetti *geoWrite Workshop* e *DeskPack Plus* siano immutate. Di esse quindi non parleremo, rimandando il lettore agli articoli "Gestire i testi in ambiente *GEOS*: *geoWrite Workshop*" apparso sul numero 4/88, e al box "*DeskPack Plus*" apparso sul numero 3/88 di *Commodore Gazette*.

DeskTop V1.5

Cominciamo con l'applicazione che ha subìto più modifiche: *deskTop*. La cosa non ci stupisce, perché *deskTop* è l'interfaccia utente tramite la quale si svolge la maggior parte delle operazioni con i dischi e i file, l'applicazione a cui si fa ricorso più spesso, e per questo anche la più soggetta a critiche. La Berkeley non si è fatta attendere, e ora *deskTop* appare sensibilmente migliorata.

Appena avviato il sistema, si nota subito che lo schermo di *deskTop*, la famosa scrivania con il bloc-notes, presenta qualche cambiamento. Il menu principale contiene due voci in più, Select e Page, e sulla sinistra in alto appare finalmente il tanto sospirato orologio; inoltre, l'icona della stampante si trova ora in basso a sinistra, agli antipodi rispetto al cestino e non si corre più il rischio di cancellare per errore quello che invece si voleva stampare. Ma basta effettuare qualche esplorazione per rendersi conto che le novità sono molte di più.

Tramite il menu Select è possibile selezionare più file contemporaneamente ed eseguire operazioni sull'intero gruppo. Per esempio, è possibile selezionare una dopo l'altra le applicazioni che si vogliono copiare nel RAM disk, e avviare la copia una volta sola. Sarà *deskTop* che estenderà l'operazione di copia a tutti i file selezionati. Ovviamente, anche la cancellazione e le altre operazioni si possono estendere a più file. La selezione può avvenire tramite le voci del menu o tramite opportune combinazioni di tasti e selezioni del mouse sulle icone. Si possono anche selezionare in un colpo solo tutti i file presenti sul disco, tutti i file contenuti nella pagina del notes aperta, o tutti i file depositati sulla scrivania. Chi ha usato una versione precedente di *deskTop* sa bene quanto sia tedioso non poter estendere operazioni come la copia e la cancellazione a una pluralità di file, e quindi questa nuova possibilità è senza dubbio una delle più significative. Sebbene non esista ancora la possibilità di redigere file batch, dato che manca sempre un'interfaccia linea comandi (siamo comunque certi che qualcuno alla Berkeley ci sta spendendo notti insonni), l'opzione Select vi si avvicina, e rende le varie sessioni di lavoro molto più veloci.

La seconda voce aggiunta al menu principale, Page, sarà accolta con favore da coloro che desiderano i propri dischi sempre ordinatissimi. Permette infatti di aggiungere una pagina bianca al notes in qualsiasi punto della sequenza, e di rimuovere altrettanto facilmente qualsiasi pagina, cancellando ovviamente tutti i file in essa contenuti.

Alcune delle altre voci sono invece state dotate di nuove opzioni. Il sotto-menu File contiene ora la voce Undo Delete, grazie alla quale è possibile recuperare l'ultimo file cancellato, a patto che non siano stati compiuti nel frattempo accessi al disco in scrittura. Il nome dell'ultimo file cancellato è visibile proprio sotto il cestino.

L'altro sotto-menu che ha subìto modifiche è Special, che ora si chiama Options. Le aggiunte servono per modificare l'ora dell'orologio, e per visualizzare sullo schermo tutte le combinazioni di tasti (hot-key) previste dal programma. Si noti a questo proposito che ora quasi tutte le operazioni selezionabili con il mouse sono attivabili anche da tastiera. Qualcuno giudica la presenza di hot-key incongruente con la filosofia user-friendly di *GEOS*, ma la nostra opinione è che un'interfaccia grafica per essere davvero efficiente deve prevedere anche l'uso della tastiera, cioè deve offrire all'utente la facoltà di scegliere il sistema interattivo che preferisce. Anche perché la tastiera rimane sempre il dispositivo più rapido per molte operazioni ripetitive.

Sempre in materia di comandi da tastiera, *deskTop V1.5* permette di scegliere una qualunque delle 18 pagine della directory, contro le prime nove consentite dalle precedenti versioni, e di selezionare le icone disposte sul notes con altre opportune combinazioni di tasti (evidentemente qualche programmatore aveva una particolare simpatia per le combinazioni di tasti).

L'orologio-datario alla destra del menu principale è quel genere di optional che tutti avevano sempre desiderato, anche se non era così importante da suscitare esplicite proteste. Individuare un disco contenente Preference Manager, togliere la protezione, inserirlo nel disk drive e caricare il desk accessory soltanto per cambiare l'ora è un'operazione che tutti si sono sempre rifiutati di fare. Adesso l'ora e la data si possono aggiornare con il semplice spostamento del mouse e finalmente acquista un senso l'aver previsto l'ora e la data per i file *GEOS* compatibili

Un'altra importante novità riguarda l'opportunità di assegnare colori permanenti alle icone dei file, in modo che si possa distinguerle sullo schermo ancora più rapidamente. Per assegnare i colori si usa Pad Color Manager, un nuovo desk accessory, che permette anche di tornare al semplice schermo monocromatico se la novità non risulta gradita. Peccato che non sia ancora possibile scegliere una particolare matrice grafica per lo sfondo (la scrivania), per evitare le bande verticali colorate che talvolta appaiono su alcuni monitor.

Anche le operazioni con i dischi e i file sono state in parte migliorate. La copia, per esempio, è diventata più rapida, e richiede all'utente meno scambi di dischi se si possiede un solo disk drive. Oltre alla consueta formattazione di un disco, è ora possibile effettuarvi la semplice cancellazione della directory e della BAM, opzione molto comoda quando un disco non contiene settori illeggibili ed è quindi inutile perdere tempo con una completa riformattazione. Inoltre, ora è possibile scambiare vicendevolmente di posto due icone sulla stessa pagina portandole semplicemente una sull'altra e premendo il pulsante del mouse.

L'ultima e più importante novità di *deskTop V1.5* è la possibilità di utilizzare tre dispositivi di memoria esterni: due drive logici, A e B, e un terzo drive logico che diventa operativo solo se viene scambiato con uno degli altri due. Per farsene un'idea più chiara, supponiamo che il drive logico A sia il disk drive 1541, che il B sia il 1571, e che il terzo drive logico sia il RAM disk. In questo caso, sullo schermo di *deskTop* appaiono ora tre icone per i drive logici, due simili (quelle effettivamente utilizzabili) e una diversa per il drive logico disponibile ma momentaneamente inattivo. Se l'utente desidera usare quest'ultimo, deve prima scambiarlo con uno degli altri due, rendendone inattivo un altro. Forse il sistema potrà sembrare complicato, ma attualmente non è possibile fare di meglio, dal momento che il Kernel di *GEOS* è in grado di "vedere" un massimo di due disk drive (si consulti la *Guida ufficiale alla programmazione di GEOS* per una descrizione completa del problema). Contrariamente a quanto affermano anche autorevoli testate americane, realizzare una normale gestione a tre disk drive non sarebbe stato per niente facile, e avrebbe richiesto un radicale intervento sullo stesso sistema operativo. Quindi, non lamentiamoci troppo, tanto più che ora, oltre ai RAM disk, è anche possibile impiegare il bistrattato disk drive 1581 (a tutto vantaggio della Commodore, oltre che degli utenti), aumentando notevolmente la quantità dei dati che si possono memorizzare sullo stesso disco.

A quanto ci risulta, queste sono le più importanti variazioni subite da *deskTop*. Sono tutte molto utili, ma ci rammarica il fatto che non sia ancora possibile selezionare i file quando vengono elencati sul notes senza le icone, limite che rende inutile evitare la raffigurazione a icone.

Configure e il disk drive 1581

Un'altra importante novità di *GEOS 2.0*, come abbiamo anticipato, è la capacità di essere configurato, tramite il programma *Configure*, per interagire anche con il disk drive 1581. Si tratta di una grande novità, che riporta in auge un disk drive molto buono, ma poco sfruttato. In effetti, la Berkeley è stata con *GEOS 128* la prima software house a riconoscere l'utilità del 1581, come è stata anche la prima a sfruttare a fondo le espansioni RAM. Con *GEOS 2.0* anche gli utenti del C-64 potranno finalmente avvalersi di 800K di memoria su dischi da 3,5", circa lo spazio che si ottiene con cinque dischi da 5,25" formattati su un solo lato. Considerando quanto siano importanti e frequenti gli accessi al disco in ambiente *GEOS*, la possibilità d'impiegare il 1581 contribuirà a rendere l'ambiente di lavoro ancora più efficiente.

Il file *Configure* è stato modificato per poter assegnare un disk drive anche al terzo drive logico previsto da *deskTop*. Non disponendo ancora del pacchetto, non ci è possibile dire di più. Possiamo solo ricordare che con l'espansione 1764 (256K) è possibile ottenere un RAM disk da 170K. Se invece si espande la 1764 a 512K, o se si dispone di una 1750, è possibile ottenere anche un RAM disk da 366K. Si noti inoltre che nessuna delle due possibilità sfrutta completamente la RAM disponibile nelle due espansioni, e GEOS può quindi usarne una parte per diverse funzioni: mantenere uno shadowed disk, aumentare la velocità dei trasferimenti di memoria nel sistema... L'utente può anche decidere di mantenere nella memoria RAM dell'espansione una copia del sistema operativo per caricarlo rapidamente dopo un reset (non si dimentichi che i RAM disk non vengono alterati durante il reset del computer, e che quindi se si riattiva *GEOS* il loro contenuto è sempre invariato).

GeoPaint

Il successo di *geoPaint*, nonostante i limiti che tutti conoscono, non è mai diminuito, soprattutto perché consente di disegnare su un intero foglio di stampa (8 x 10 pollici). Si sentiva però la forte esigenza di alcuni miglioramenti e ora, con *GEOS 2.0*, viene fornita una nuova versione del programma.

Le funzioni di cut & paste sono state migliorate e non consentono solo di "importare" un'immagine ma anche di sottoporla a operazioni di scaling, distorsione, e smussamento (come quelle che abbiamo descritto nell'articolo dedicato a *geoPublish* nel numero 2/88). Inoltre, è possibile muovere l'immagine importata e cambiarne le dimensioni con due semplici gadget, e decidere se lo sfondo deve rimanere visibile o no (effetto trasparenza).

L'opzione Spray è stata modificata per consentire di riprodurre una matrice grafica visualizzandone solo i punti neri, solo i punti bianchi, o entrambi come nelle precedenti versioni. Anche gli altri tool di disegno sono stati considerevolmente migliorati, specialmente sotto il profilo della velocità. In particolare, è diventato possibile disegnare oltre ai cerchi anche le ellissi e le poligonali.

Per quanto riguarda gli spostamenti del puntatore, sono ora possibili regolazioni fini con i tasti cursore. Il controllo fine è disponibile anche con il modo Pixel Edit.

Infine, ci risulta che ora sia possibile attivare una griglia di riferimento (di dimensioni purtroppo inalterabili), bloccare il riempimento di aree, limitare l'operatività dei tool all'interno di particolari aree, disegnare rette con inclinazioni multiple di 45°. Queste sono in sostanza le innovazioni principali, alle quali daremo ampio spazio in un prossimo articolo.

Le altre innovazioni

Lo spelling checker *geoSpell* è praticamente stato riscritto. Anche se prevede solo un vocabolario di parole inglesi, ed è quindi di scarso interesse per il mercato italiano, uno dei suoi più grandi limiti era la lentezza nella ricerca. La Berkeley afferma che la nuova versione inserita in *GEOS 2.0* è più veloce della precedente di circa il 38 per cento, un dato che verificheremo al più presto.

Oltre al nuovo desk accessory Pad Color Manager, *GEOS 2.0* porta anche diversi miglioramenti in Photo e Text Manager, che permettono ora di assegnare nomi alle immagini scrap e ai testi che gestiscono, facilitandone l'individuazione. Con le versioni precedenti, infatti, occorre sfogliare l'intero album per trovare una particolare immagine, un'operazione talvolta molto lunga. Ora appare un box di dialogo tramite il quale l'utente sceglie facilmente il file scrap che desidera.

Anche i desk accessory Calculator e Notepad sono stati migliorati. Calculator consente di copiare i risultati ottenuti all'interno di file scrap che si possono poi leggere da *geoWrite*, e da qualunque altra applicazione dotata delle funzioni di cut & paste. Notepad, a sua volta, è ora in grado aggiungere e cancellare pagine del suo piccolo bloc-notes. L'unico problema riguardo a queste nuove opzioni nei due

desk accessory sembra essere che il solo modo per azionarle è quello di utilizzare opportune combinazioni di tasti. La mancanza di menu si può spiegare soltanto come una scelta strategica mirata a valorizzare la necessità del manuale, dal momento che inserirli non sarebbe certo difficile. Lo svantaggio è la difficoltà di ricordarsi ogni volta la combinazione giusta, se non si consulta il manuale (cosa che infrange la regola aurea delle interfacce grafiche user-friendly: mai costringere l'utente a consultare il manuale).

Conclusioni

Sebbene la nostra non sia una "prova software" nel vero senso della parola, su *GEOS 2.0* sono emerse novità di rilievo che non mancheranno d'incuriosire gli utenti. Anche se non possiamo ancora esprimere valutazioni, ci fa piacere notare che la Berkeley rinnova costantemente la propria produzione mantenendo vivo l'interesse verso un prodotto sempre all'avanguardia. Alla Berkeley spetta anche il merito di aver creato uno standard per tutta una serie di applicazioni, un fatto unico nella vita del venerando C-64.

GEOCHART

BUSINESS GRAPHICS IN AMBIENTE GEOS

Quando abbiamo reperito i dati numerici che descrivono l'andamento di un fenomeno, dobbiamo ancora sottoporli ad analisi statistiche per individuare simmetrie, discordanze, anomalie, e trarre conclusioni di massima che descrivano nel modo migliore il fenomeno nel suo complesso. Quest'analisi può essere condotta elaborando i dati originali, scoprendo per esempio che un certo parametro è cresciuto con un ritmo medio di x incrementi unitari per unità di tempo in un certo periodo, e che successivamente ha assunto un andamento stabile... ma ci troveremmo in mano altri dati numerici, più raffinati e significativi dei precedenti, ma sempre oscuri e poco familiari.

L'alternativa è l'analisi qualitativa, più approssimata ma molto più immediata, e per svolgerla occorre svincolarsi dai dati in forma numerica individuando altri modi di rappresentazione (non è un caso che allo studente di analisi sia richiesto lo studio numerico delle funzioni, ma soprattutto la capacità di descrivere il loro andamento di massima su un sistema di assi cartesiani).

Uno dei settori che maggiormente richiede analisi qualitative dei fenomeni è quello economico-commerciale, per il quale è nata la business graphics, la rappresentazione grafica di dati economici, statistici e commerciali. Ovviamente, anche gli andamenti di fenomeni più strettamente scientifici vanno talvolta rappresentati graficamente per evidenziare i risultati, ma ormai il termine business graphics è talmente diffuso che lo si adotta anche in questi casi.

Nel mondo di *GEOS*, che nei suoi limiti cerca di soddisfare esigenze di carattere professionale, era molto atteso un programma di business graphics. Anche perché, com'è noto, in questo ambiente è estremamente facile scambiare file fra applicazioni diverse: per esempio, ottenere un istogramma in un file scrap su disco e importarlo all'interno di una pagina composta con *geoPublish* o all'interno di un testo redatto con *geoWrite*.

La risposta della Berkeley alle esigenze degli utenti in materia di business graphics è *geoChart*, un'applicazione che permette di rappresentare graficamente insiemi di dati utilizzando ben nove formati diversi: le classiche torte, aree, barre, punti, linee, colonne, barre uniche, grafici a punti e a linee. A questa applicazione dedicheremo al più presto una dettagliata prova, ma qualche accenno possiamo farlo già in questa sede.

Anzitutto, *geoChart* può funzionare sia con *GEOS 64* sia con *GEOS 128*. Quando viene mandato in esecuzione, *geoChart* visualizza sullo schermo un istogramma dimostrativo, che purtroppo sembra impossibile da evitare (ci vogliono circa 15 secondi perché sia completato). Per introdurre i propri dati occorre importare un file scrap di testo che può essere generato da *geoWrite*, *geoCalc*, *geoFile*, o dalla nuova versione del desk accessory Notepad (contenuto sullo stesso disco di *geoChart*). Dal momento che su un disco può risiedere un solo file scrap alla volta, è consigliabile servirsi del desk accessory Text Manager per catalogarne diversi. A prescindere dalla provenienza, questo file scrap deve riportare i dati da rappresentare organizzati secondo una particolare disposizione che permette a *geoChart* d'individuare senza ambiguità l'intestazione dell'immagine che creerà, i dati da rappresentare, la loro classificazione, e i raggruppamenti previsti. Quando i dati sono stati importati, *geoChart* è pronto per ricevere istruzioni dall'utente e creare la rappresentazione desiderata.

Una volta che i dati sono in memoria, si può continuare a cambiare la loro rappresentazione senza doverli ricaricare, sebbene ogni intervento radicale sia abbastanza oneroso in termini di tempo. In ogni momento si può cambiare il tipo di rappresentazione, la disposizione degli eventuali assi (le coordinate possono anche essere negative), le matrici grafiche da utilizzare per visualizzare le barre e le torte, la forma dei punti e il tipo di linea nei grafici, la trasparenza nel caso di aree sovrapposte, il tipo di carattere per legende e commenti (si possono usare tutte le fonti-carattere previste da *GEOS*, comprese le 53 del pacchetto *FontPack Plus*), nonché gli stili di scrittura, le posizioni dei testi, le dimensioni dei grafici, la rappresentazione degli indici numerici che appaiono sugli assi cartesiani scegliendo fra 14 possibilità (come per esempio numeri preceduti dal simbolo di dollaro o seguiti dal simbolo di percentuale). Inoltre, la mole di dati rappresentabili può raggiungere gli 80 valori, suddivisi in quattro insiemi fondamentali da 20 valori ciascuno.

Quando la rappresentazione ha raggiunto una forma definitiva, si può procedere a stamparla direttamente e salvarla in un file scrap di dati grafici. Questo file può poi essere trasferito a qualsiasi applicazione in grado di leggerlo e inserirlo all'interno di testi, pagine o disegni, come *geoWrite*, *geoPublish*, *geoPaint*. *GeoChart* prevede diversi tipi di file scrap, a seconda dell'applicazione che dovrà riceverli. Le differenze riguardano soprattutto le dimensioni dell'immagine.

In definitiva, *geoChart* sembra un prodotto interessante e utile, soprattutto per coloro che già impiegano programmi come *geoCalc* e *geoFile*. Le sue applicazioni possono essere tantissime. Si pensi a quando si vorrebbe osservare su un grafico l'andamento del proprio stipendio in rapporto alle variazioni di prezzo dei beni di prima necessità negli ultimi cinque anni, oppure le fluttuazioni della bolletta telefonica in rapporto ai vari periodi dell'anno, o ancora si vorrebbero confrontare con un istogramma le spese di due aziende simili. Le necessità possono essere le più disparate, e *geoChart* può diventare lo strumento ideale per soddisfarle ■

UN C-128 A CACCIA DI ECLISSI

Terzo listato della serie "astronomia": dopo aver individuato le posizioni dei pianeti e delle comete, puntiamo ora il nostro telescopio a otto bit verso il cielo alla ricerca delle eclissi di Sole e di Luna

di Marco Menichelli

Il tema dell'astronomia, già introdotto con il numero 1/88 di *Commodore Gazette*, prosegue con il programma *Eclissi* che calcola e visualizza velocemente gli eventi celesti più affascinanti: le eclissi di Sole e di Luna.

Oggi tutti sappiamo come avviene un'eclisse, quindi non annoieremo i lettori cercando di spiegare il meccanismo dei reciproci movimenti di Terra e Luna che consente il verificarsi di questi fenomeni. Passeremo subito a illustrare invece il funzionamento del programma, in modo che il suo corretto impiego possa soddisfare coloro che vogliono ricostruire sul monitor del computer un'eclisse avvenuta nell'anno 1300, oppure analizzare un'eclisse che avverrà nel 2500.

Il programma si articola in due fasi. La prima calcola e visualizza le informazioni relative alle eclissi di un determinato anno, e mostra sull'uscita RGBI a 80 colonne in quale istante si verificano. La seconda fase calcola le posizioni e le dimensioni in scala della Luna e del Sole (per le eclissi di Sole), oppure della Luna, dell'ombra e della penombra della Terra (per le eclissi di Luna). Le immagini relative al fenomeno vengono visualizzate sul monitor connesso all'uscita delle 40 colonne, considerando una finestra di cielo graduata secondo le coordinate altazimutali relative all'orizzonte del luogo di osservazione.

Dopo aver mandato in esecuzione il programma, appare il menu con la possibilità di scegliere tra quattro programmi di astronomia. I primi due sono stati pubblicati nei numeri 1/88 e 4/88 di *Commodore Gazette* e l'ultimo, *Visibilità*, verrà pubblicato prossimamente.

Per avviare *Eclissi* è necessario premere il tasto Return e poi scegliere fra eclissi di Sole o eclissi di Luna. Una volta comunicata la decisione è possibile inserire l'anno relativo alla ricerca delle eclissi, compreso fra il 1000 e il 3000.

In questo programma, tutti i tempi relativi agli eventi vengono espressi in T.E., che sta per Tempo delle Effemeridi. Prima di proseguire è necessario chiarire il significato e il valore di questa unità di misura. Sui programmi pubblicati precedentemente abbiamo definito il Tempo Universale (T.U.) come il tempo relativo al meridiano di Greenwich (un'ora meno di quello segnato dai nostri orologi). Dal momento che la velocità di rotazione della Terra sul suo asse non è uniforme ma aumenta e diminuisce in modo irregolare, anche il T.U., legato a questo moto, non può essere un tempo uniforme. Per i calcoli astronomici che coinvolgono epoche distanti alcuni secoli dal 1900 è invece necessario fare riferimento a un tempo uniforme, appunto il Tempo delle Effemeridi.

La differenza fra T.E. e T.U. negli anni vicini al 1900 oscilla intorno al valore di più o meno qualche decimo di minuto. Per l'anno 1989 sta raggiungendo il valore di un minuto.

Il programma visualizza automaticamente la differenza T.E. – T.U. quando questa supera il minuto. È necessario sottrarre algebricamente dal T.E. questa differenza per conoscere il T.U. dell'evento scelto.

Eclissi di Sole

Se, per esempio, desideriamo conoscere i dati relativi alle eclissi di Sole avvenute nell'anno 1239, risponderemo "S" alla prima richiesta e "1239" alla seconda. Dopo un'attesa di alcuni secondi potremo leggere sul monitor a 80 colonne i dati che seguono.

T.U. = T.E. – (14 minuti)

TIPO	GRANDEZZA	VISIBILITÀ	DATA	T.E.: H	M
TOTALE CENTR.	1.000	NORD	3/6/1239	12	1.8
ANULARE CENTR.	1.000	SUD	27/11/1239	4	33.7

Ciò significa che nel 1239 si verificarono due eclissi di Sole. La prima, totale, visibile dall'emisfero settentrionale, avvenne il 3 giugno alle ore 12.02 circa (Tempo delle Effemeridi). La seconda, anulare, avvenne il 27 novembre alle ore 4.34 circa di T.E. e fu visibile dall'emisfero meridionale.

Non ci rimane che scegliere quale delle due vogliamo visualizzare sul monitor dedicato alla grafica. Premendo il tasto CRSR DOWN sarà possibile fare la nostra scelta. Premiamo il tasto Return per confermare la visualizzazione di quella del 3 giugno. Ora dovremo fornire al computer quattro dati necessari a definire i parametri per il calcolo delle posizioni del Sole e della Luna relative all'orizzonte locale.

Il primo input si riferisce all'ora dell'osservazione, sempre in T.E., da immettersi nella forma HHMM, ore e minuti.

Il secondo e il terzo input sono relativi alla longitudine e alla latitudine del luogo di osservazione e, com'è già noto a chi ci sta seguendo da qualche mese, vanno scritti col loro valore in gradi e decimi di grado. Per la longitudine terrestre è possibile inserire una cifra compresa fra 180.0 e –180.0 ricordando che le longitudini positive si riferiscono alle zone della Terra che si trovano a est del meridiano di Greenwich, le altre, negative, alle zone che si trovano a ovest dello stesso meridiano. La latitudine terrestre, invece, può spaziare fra 89.9 e 0 per l'emisfero nord e fra 0 e –89.9 per quello sud.

Il quarto input richiede l'altezza in metri sul livello del mare del luogo di osservazione.

Se al momento dei quattro input non si immette alcuna cifra e si preme semplicemente il tasto Return si ottiene la visualizzazione dell'eclisse calcolata con i valori di default. Premiamo quattro volte Return ottenendo i valori 12H 1M per il T.E., 11.3 per la longitudine, 43.8 per la latitudine e 300 per l'altezza in metri. Gli ultimi tre dati sono relativi alla città di Fiesole (Firenze e dintorni) come zona di osservazione.

Dopo alcuni secondi assisteremo al disegno sul monitor del grafico relativo a una porzione di cielo, dove due cerchi rappresentano le figure in scala del Sole e della Luna. Per primo e in posizione centrale appare il cerchio del Sole, poi viene tracciato il disco della Luna. A sinistra e sotto il rettangolo raffigurante la porzione di cielo notiamo alcuni valori numerici che definiscono le coordinate altazimutali del rettangolo stesso. Nel caso specifico noteremo dal basso verso l'alto i valori in altezza di 65.7, 66.9 e 68.1. In fondo i valori dell'azimut compresi fra 208.4 e 211.9, tutti espressi in gradi e decimi di grado. La Luna, decentrata, sfiora appena il disco del Sole. Così, alle 12.01 (T.E.), da Fiesole si vedeva l'inizio dell'eclisse. Controlliamo ora se ai genitori di Dante Alighieri fu possibile osservare questa eclisse come totale. Posticipiamo di una trentina di minuti la visualizzazione dell'evento: si prema Return, si introduca come T.E. le ore 12.37 e si impostino le stesse coordinate del luogo di osservazione. Questa volta abbiamo fatto centro: alle 12.37 di T.E. il disco della Luna aveva coperto completamente quello del Sole provocando così una spettacolare eclisse totale di Sole. Le coordinate azimutali del Sole e della Luna erano di 226.9 gradi, contati a partire da nord verso est, e la loro altezza sull'orizzonte valeva 62.9 gradi.

Verificate l'esattezza dei dati del vostro programma confrontandoli con questi ultimi.

I lettori che abitano in altre città provino a introdurre i dati relativi alle loro coordinate geografiche locali. Lo spostamento del punto di osservazione farà variare sensibilmente sul cielo locale la posizione del Sole e della Luna.

Le coordinate di alcune delle principali città italiane sono state pubblicate sul numero 1/88 di *Commodore Gazette*, nel corso dell'articolo "Osserviamo gli astri attraverso il C-128".

Proviamo ora a ripetere il procedimento con l'anno 1961 selezionando l'eclisse totale avvenuta il 15 febbraio sull'emisfero nord alle ore 8 e 23.8 di T.E., anch'essa visibile dall'Italia. Immettiamo un valore di T.E. pari alle 7.37. Se confermiamo Fiesole come luogo di osservazione, la Luna deve

Eclisse anulare del 30-5-1984 alle ore 20 e 10 minuti circa di Tempo Civile corrispondente alle 18 e 10 minuti di T.E.

coprire il disco del Sole, e i dati relativi all'altezza e all'azimut dell'astro oscurato devono valere rispettivamente 12.7 e 122 gradi.

Eclissi di Luna

Queste eclissi, meno spettacolari di quelle di Sole, hanno il vantaggio di poter essere osservate da una più ampia superficie del nostro pianeta. Pertanto qualsiasi osservatore che si trova sull'opportuno meridiano, sia nell'emisfero nord sia nell'emisfero sud, potrà godere, entro certi limiti, dello stesso spettacolo. Verranno talvolta delusi solo coloro che abitano oltre i due circoli polari.

Per analizzare le eclissi di Luna osserveremo lo stesso procedimento, salvo quanto concerne la scelta iniziale che indirizzeremo verso l'apposito algoritmo Lunare premendo il tasto "L". Proviamo a effettuare la ricerca nell'anno 1986. Selezioniamo ora l'eclisse totale avvenuta il 17 ottobre alle ore 19 e 18.5 minuti di T.E. (fase centrale dell'eclisse). Confermiamo i dati di default premendo tre volte Return. Il programma visualizza i dati circostanziali dell'eclisse di Luna nella sua fase centrale, con gli istanti di T.E. riportati nella tavola a fondo pagina. Dopo alcuni secondi si ottiene la descrizione grafica dell'eclisse, nella quale le zone dell'ombra e della penombra provocate dalla Terra e il piccolo disco della Luna vengono rappresentate da tre cerchi. Il primo, quello di raggio maggiore, rappresenta il limite della penombra. Il secondo, più piccolo e concentrico al primo, rappresenta il limite dell'ombra. L'altezza del centro dell'ombra sull'orizzonte letta sul grafico vale 31.1 gradi, mentre il suo azimut risulta 108.4 gradi. Le coordinate altazimutali del centro della Luna, rappresentata dall'ultimo cerchio tracciato, valgono approssimativamente H = 31.5 e A = 107.8 gradi.

È possibile visualizzare eclissi anche quando la Luna è già tramontata o prima del suo sorgere sull'orizzonte locale. In questi casi i valori relativi all'altezza sono negativi e l'informazione lampeggiante "Invisibile da queste coordinate a quest'ora" avverte l'astrofilo disattento. Il medesimo risultato si ottiene quando si va alla ricerca di un'eclisse di Sole verificatasi dopo il tramonto o prima del sorgere dell'astro.

Eclisse parziale del 15-12-82 alle ore 9 e 15 minuti circa di Tempo Civile corrispondenti alle 8 e 15 minuti di T.E.

Quantità delle eclissi in un anno

Al fine di evitare infruttuose ricerche ai neoastrofili interessati a questo programma, è opportuno chiarire che nell'arco di un anno possono aver luogo da due a cinque eclissi di Sole e da zero a tre eclissi di Luna totali.

Le eclissi non potranno comunque essere complessivamente più di sette o meno di due, nel qual caso saranno esclusivamente eclissi di Sole (non sono considerate le eclissi penombrali di Luna). In sintesi potranno aversi al massimo in un anno cinque eclissi di Sole e due di Luna, oppure quattro di Sole e tre di Luna.

Qualità dei risultati

Il T.E. calcolato dal computer nella prima parte del programma si riferisce sempre alla fase centrale dell'eclisse; è quindi valido soltanto per una piccola zona sulla superficie terrestre. Inoltre può differire di uno o due minuti rispetto ai valori reali. Questa differenza è soggetta ad aumentare per le ricerche che si estendono oltre il secolo, fino a raggiungere nei casi estremi anche i 30 minuti d'errore: risultati più che soddisfacenti per l'uso amatoriale a cui è destinato il programma.

Durante la ricerca delle eclissi, a causa del numero di funzioni trigonometriche richieste e dei decimali in gioco, è possibile che il programma scambi un'eclisse totale per una parziale, o più semplicemente che non la identifichi quando si trova al suo limite.

I lettori che sono già in possesso dei listati relativi ai programmi *Astronomia* e *Comete* dovranno salvare il programma *Eclissi* sullo stesso disco. Perché *Eclissi* entri a far parte integrante degli altri due programmi summenzionati sarà sufficiente rendere attiva la linea 510 di *Astronomia* e la linea 630 di *Comete* togliendo il comando REM e gli asterischi.

La prossima eclissi totale di Sole visibile anche dall'Italia avverrà nell'anno 2081. Considerato che saremo in pochi a potervi assistere, perché non cercare di riprodurla sul nostro monitor?

Inizio fase parziale	17H 30M	**Fine fase parziale**	21H 6M
Inizio fase totale	18H 41M	**Fine fase totale**	19H 55M

Listato: *Eclissi C-128*

```
10 REM ************************
   *********
20 REM **
         **
30 REM **       ECLISSI     C=12
   8       **
40 REM **
         **
50 REM **    (C) MARCO MENICHEL
   LI      **
60 REM **
         **
70 REM **         FIESOLE (FI)
         **
80 REM **
         **
90 REM ************************
   *********
100 :
110 SCNCLR:IF RGR(0)<>5 THEN CH
    AR,0,12,"RATTENZIONE !■ INP
    UT & DATI SU USCITA RGBI":S
    LEEP 2
120 BANK 15:FAST:GRAPHIC 5,1:WI
    NDOW 0,0,79,24
130 CHAR,20,20,"GRAFICA VISIBIL
    E SU MONITOR A 40 COLONNE"
140 CHAR,29,10,"┌────────────
    ────┐":CHAR,29,11,"|
    ECLISSI        |"
150 CHAR,29,12,"| (C) M. MENICH
    ELLI |":CHAR,29,13,"└──────
    ──────────┘":SLEEP1
160 FAST:SCNCLR:WINDOW 0,0,79,2
    4,1:CLR:COLOR 0,1:COLOR 4,1
    :COLOR 1,2:GRAPHIC 1:GRAPHI
    C 0:GRAPHIC 5:CHAR,18,0,"AS
    TRONOMIA - (C) MARCO MENICH
    ELLI - FIESOLE",1
170 :
180 REM ** MENU PRINCIPALE, DEF
    INIZIONE DELLE VARIABILI E
    DELLE FUNZIONI **
190 :
200 DEF FNA(X)=X*π/180:DEF FNB(
    X)=X*180/π:DEF FNTA(X)=ATN(
    X/SQR(-X*X+1))
210 DEF FNC(X)=(X/2/π-INT(X/2/π
    ))*2*π:DEF FNE(X)=X-INT(X/3
    60)*360:DEF FNF(X)=X-INT(X/
    24)*24
220 JK$="████████████"
230 TP$="####################
    ######  #.###           ####
          ##/##/####       ## #
    #.#"
240 HM$="R##H ##M■<":LE$="R-###
    .#■":LK$="R-##.#■":HU$="R##
    ####■"
250 FOR X=1 TO 75:SC$=SC$+" ":N
    EXT:SP$=LEFT$(SC$,9)
260 XN=2415020.0
270 N=1:CHAR ,25,16,"SCEGLI CON
     CRSR E PREMI RETURN",1
280 POKE 208,0:DO WHILE A$<>CHR
    $(13)
290 CHAR ,29,5,"        ECLISSI
          ",-(N=1)
300 CHAR ,29,7," EFFEMERIDI SIS
    T.SOL. ",-(N=2)
310 CHAR ,29,9," EFFEMERIDI
    COMETE ",-(N=3)
320 CHAR ,29,11,"        VISIBILI
    TA'        ",-(N=4)
330 CHAR ,29,13,"          FINE
          ",-(N=5)
340 GET A$:IF A$=CHR$(17)THEN N
    =N+1:N=N+(N>5)*5
350 IF A$=CHR$(145) THEN N=N-1:
    N=N-(N=0)*5
360 LOOP
370 :
380 ON N GOTO 4010,420,430,440,
    480
390 :
400 REM ** CARICA UNO DEGLI ALT
    RI PROGRAMMI DI ASTRONOMIA
    **
410 :
420 RUN "ASTRONOMIA"
430 RUN "COMETE"
440 REM ** RUN "VISIBILITA" **
450 :
460 REM ***** FINE *****
470 :
480 WINDOW 0,0,79,24:SLOW:SCNCL
    R:END
490 :
500 REM ** SUBROUTINE DI INPUT
    CONTROLLATO **
510 :
520 JK=0:UZ$=LEFT$(SP$,UZ):POKE
     208,0:A$="":DO WHILE A$<>C
    HR$(13)
530 IF JK=UZ THEN 560
540 IF A$=CHR$(44) OR A$=CHR$(4
    7) THEN 580
550 IF ASC(A$)>MN AND ASC(A$)<M
    X THEN UZ$=RIGHT$(UZ$+A$,UZ
    ):JK=JK+1
560 IF A$=CHR$(157) OR A$=CHR$(
    20) THEN UZ$=" "+LEFT$(UZ$,
    UZ-1):JK=JK-1
570 PRINT UZ$;"<";LEFT$(JK$,UZ+
    1);
580 GET A$:LOOP
590 RETURN
600 :
610 :
620 REM ** COORDINATE E LIVELLO
     SUL MARE DEL LUOGO DI OSSE
    RVAZIONE **
630 :
640 LE=11.3:LK=43.8:HU=300: REM
     ** FIESOLE **
650 :
660 CHAR,30,0,SP$:CHAR,3,6,SC$:
    CHAR,3,8,SC$
670 :
680 REM ** VARIAZIONE EVENTUALE
     DEL T.E. E/O DELLE COORDIN
    ATE DI OSSERVAZIONE **
690 :
700 CHAR,0,0,"ORE E MINUTI DI T
    .E.  (HHMM) >":MN=47:MX=58:
    UZ=4:GOSUB 520:IF UZ$="
    " THEN IU=1:GOTO 740:ELSE H
    Y=VAL(MID$(UZ$,1,2)):MY=VAL
    (MID$(UZ$,3))
710 IF HY>23 OR MY>59 THEN 700:
    ELSE HH=HY+MY/60:HX=HH/24:J
    J=JJ(N)
720 IF JJ-INT(JJ)=>.5 THEN JJ=I
    NT(JJ)+.5:ELSE JJ=INT(JJ)-.
    5
730 JJ=JJ+HX
740 CHAR,30,0:PRINT USING HM$;I
    NT(HH),INT((HH-INT(HH))*60)
750 IF XC THEN BEGIN
760 CHAR,40,0,"LONGITUDINE >":M
    N=43:MX=58:UZ=6:GOSUB 520:I
    F UZ$="       " THEN 780
770 IF VAL(UZ$)>180 OR VAL(UZ$)
    <-180 THEN 760:ELSE LE=VAL(
    UZ$)
780 CHAR,53,0:PRINT USING LE$;L
    E
790 CHAR,61,0,"LATITUDINE >":UZ
    =5:GOSUB 520:IF UZ$="      "
     THEN 810
800 IF VAL(UZ$)>89.9 OR VAL(UZ$
    )<-89.9 THEN 790:ELSE LK=VA
    L(UZ$)
810 CHAR,73,0:PRINT USING LK$;L
    K
820 :
830 REM ** CIRCOSTANZE DELL'ECL
    ISSE DI LUNA **
840 :
850 CHAR,0,2,"INIZIO FASE PARZI
    ALE IN T.E. >":PRINT USING
    HM$;INT(H1),INT((H1-INT(H1)
    )*60)
860 CHAR,41,2,"FINE  FASE  PARZ
    IALE IN T.E. >":PRINT USING
     HM$;INT(H4),INT((H4-INT(H4
    ))*60)
870 IF XC=2 THEN BEGIN
880 CHAR,0,4,"INIZIO FASE   TOT
    ALE IN T.E. >":PRINT USING
    HM$;INT(H2),INT((H2-INT(H2)
    )*60)
890 CHAR,41,4,"FINE   FASE   TO
    TALE IN T.E. >":PRINT USING
     HM$;INT(H3),INT((H3-INT(H3
    ))*60)
900 BEND
910 BEND:GOTO 1000
920 CHAR,0,2,"LONGITUDINE OSSER
    VATORE       >":MN=43:MX=58:
    UZ=6:GOSUB 520:IF UZ$="
      " THEN 940
930 IF VAL(UZ$)>180 OR VAL(UZ$)
    <-180 THEN 920:ELSE LE=VAL(
    UZ$)
940 CHAR,30,2:PRINT USING LE$;L
    E
950 CHAR,42,2,"LATITUDINE  OSSE
    RVATORE        >":UZ=5:GOSUB
     520:IF UZ$="      " THEN 97
    0
960 IF VAL(UZ$)>89.9 OR VAL(UZ$
    )<-89.9 THEN 950:ELSE LK=VA
    L(UZ$)
970 CHAR,73,2:PRINT USING LK$;L
    K
980 CHAR,16,4,"ALTEZZA  SUL LIV
    ELLO DEL MARE IN METRI >":M
    N=47:MX=58:UZ=6:GOSUB 520:I
    F UZ$<>"       " THEN HU=VAL
    (UZ$)
990 CHAR,56,4:PRINT USING HU$;H
    U
1000 CHAR,30,6,"A T T E N D I .
     ....",1
1010 LO=FNA(LE):LA=FNA(LK):IF I
     U=1 AND FC=1 GOTO 2750
1020 :
1030 REM ** CALCOLO DELLA POSIZ
     IONE DEL SOLE E DELLA LUNA
      PER L'ISTANTE DELL'ECLISS
     E **
1040 :
1050 T=(JJ-XN)/36525:FC=1
1060 :
1070 REM ***** LONGITUDINE MEDI
     A DEL SOLE ****
1080 :
1090 LS=279.69668+(36000.76892+
     .0003025*T)*T:LS=FNA(LS)
1100 :
1110 REM *** ANOMALIA MEDIA DEL
      SOLE *****
1120 :
1130 MS=358.475833+((-.0000033*
     T-.000150)*T+35999.04975)*
     T:MS=FNA(MS)
1140 :
1150 REM **** LONGITUDINE MEDIA
      DELLA LUNA ****
1160 :
1170 LL=270.434164+((.0000019*T
     -.001133)*T+481267.8831)*T
     :LL=FNA(LL)
1180 :
1190 REM ***** ANOMALIA MEDIA D
     ELLA LUNA *****
1200 :
1210 ML=296.104608+((.0000144*T
     +.009192)*T+477198.8491)*T
     :ML=FNA(ML)
```

```
1220 :
1230 REM **** LONGIT. NODO ASCE
     NDENTE DELLA LUNA ****
1240 :
1250 AN=259.183275+((.0000022*T
     +.002078)*T-1934.1420)*T:A
     N=FNA(AN)
1260 :
1270 REM ** CALCOLO DI DELTA/FI
      E DELTA/E PER LA CORREZIO
     NE IN NUTAZIONE **
1280 :
1290 DF=(17.2327+.01737*T)*SIN(
     AN)-(1.2729+.00013*T)*SIN(
     2*LS)+.2088*SIN(2*AN)-.203
     7*SIN(2*LL)+(.1261-.00031*
     T)*SIN(MS)+.0675*SIN(ML)
1300 DF=DF-(.0497-.00012*T)*SIN
     (2*LS+MS)-.0342*SIN(2*LL-A
     N)-.0261*SIN(2*LL+ML)+.021
     4*SIN(2*LS-MS)-.0149*SIN(2
     *LS-2*LL+ML)+.0124*SIN(2*L
     S-AN)+.0114*SIN(2*LL-ML)
1310 DF=FNA(DF/3600)
1320 DE=+(9.2100+.00091*T)*COS(
     AN)+(.5522-.00029*T)*COS(2
     *LS)-.0904*COS(2*AN)+.0884
     *COS(2*LL)+.0216*COS(2*LS+
     MS)+.0183*COS(2*LL-AN)+.01
     13*COS(2*LL+ML)
1330 DE=DE-.0093*COS(2*LS-MS)-.
     0066*COS(2*LS-AN)
1340 DE=FNA(DE/3600)
1350 :
1360 REM *** ECCENTRICITA' DELL
     'ORBITA TERRESTRE **
1370 :
1380 EC=.01675104+(-.000000126*
     T-.0000418)*T
1390 :
1400 REM **** EQUAZIONE DI KEPL
     ERO ***
1410 :
1420 KK=FNA(.000001):E=MS
1430 TE=E-EC*SIN(E)-MS
1440 IF ABS(TE)<=KK GOTO 1500
1450 E=E-(TE/(1-EC*COS(E)))
1460 GOTO 1430
1470 :
1480 REM ** ANOMALIA VERA **
1490 :
1500 V=2*ATN(TAN(E/2)*SQR((1+EC
     )/(1-EC)))
1510 :
1520 REM ** DISTANZA SOLE-TERRA
      IN U.A. **
1530 :
1540 R=1.00000023*(1-EC*COS(E))
1550 :
1560 REM ** LONGIT. VERA DEL SO
     LE **
1570 :
1580 LV=LS+V-MS
1590 :
1600 REM *** TERMINI CORRETTIVI
      PER LA LONGITUDINE DEL SO
     LE **
1610 :
1620 AS=FNA(153.23+22518.7541*T
     ):BS=FNA(216.57+45037.5082
     *T):CS=FNA(312.69+32964.35
     77*T):GS=FNA(350.74+445267
     .1142*T-.00144*T*T)
1630 ES=FNA(231.19+20.20*T):HS=
     FNA(353.40+65928.7155*T)
1640 LV=LV+FNA(.00134)*COS(AS)+
     FNA(.00154)*COS(BS)+FNA(.0
     0200)*COS(CS)+FNA(.00179)*
     SIN(GS)+FNA(.00178)*SIN(ES
     )
1650 :
1660 REM ** CORREZIONE PER IL R
     AGGIO **
1670 :
1680 R=R+.00000543*SIN(AS)+.000
     01575*SIN(BS)+.00001627*SI
     N(CS)+.00003076*COS(GS)+.0
     0000927*SIN(HS)
1690 :
1700 REM ** OBLIQUITA' DELL'ECL
     ITTICA RISPETTO ALL'EQUATO
     RE TERRESTRE **
1710 :
1720 ET=FNA(23.452294+((.000000
     503*T-.00000164)*T-.013012
     5)*T)
1730 :
1740 REM *** CALCOLO DELLA DECL
     INAZIONE GEOCENTRICA DEL S
     OLE **
1750 :
1760 DC=FNTA(SIN(ET)*SIN(LV))
1770 :
1780 REM *** CALCOLO DELL'ASCEN
     SIONE RETTA GEOCENTRICA DE
     L SOLE ***
1790 :
1800 A7=COS(ET)*SIN(LV):A8=COS(
     LV)
1810 IF A8=0 THEN AR=π/2:GOTO 1
     830
1820 AR=ATN(A7/A8)-π*(A7>0 AND
     A8<0)-π*(A7<0 AND A8<0)-2*
     π*(A7<0 AND A8>0)
1830 IF AZ THEN AR=FNC(AR+π):DC
     =-DC
1840 :
1850 REM *** CORREZIONE DELL' A
     .R. E DECLINAZIONE DEL SOL
     E PER NUTAZIONE E ABERRAZI
     ONE ***
1860 :
1870 DN=(COS(ET)+SIN(ET)*SIN(AR
     )*TAN(DC))*DF-COS(AR)*TAN(
     DC)*DE
1880 DD=SIN(ET)*COS(AR)*DF+SIN(
     AR)*DE
1890 D1=FNA(-20.49/3600)*(COS(A
     R)*COS(LV)*COS(ET)+SIN(AR)
     *SIN(LV))/COS(DC)
1900 D2=FNA(-20.49/3600)*(COS(L
     V)*COS(ET)*(TAN(ET)*COS(DC
     )-SIN(AR)*SIN(DC))+COS(AR)
     *SIN(DC)*SIN(LV))
1910 AR=AR+DN+D1:DC=DC+DD+D2:RE
     M ** A.R. E DECL. APPARENT
     I DEL SOLE **
1920 :
1930 REM ** ELONGAZIONE MEDIA D
     ELLA LUNA **
1940 :
1950 DL=350.737486+((.0000019*T
     -.001436)*T+445267.1142)*T
     :DL=FNA(DL)
1960 :
1970 REM ** DISTANZA MEDIA DELL
     A LUNA DAL NODO ASCENDENTE
      **
1980 :
1990 FL=11.250889+((-.0000003*T
     -.003211)*T+483202.0251)*T
     :FL=FNA(FL)
2000 :
2010 REM ** TERMINI CORRETTIVI
     PER IL CALCOLO DELLA LONGI
     TUDINE LUNARE **
2020 :
2030 TC=SIN(FNA(51.2+20.2*T))
2040 SA=SIN(AN)
2050 C1=FNA(.000233*TC):C2=FNA(
     -.001778*TC):C3=FNA(.00081
     7*TC):C4=FNA(.002011*TC)
2060 CT=FNA(.003964*SIN(FNA(346
     .560+(-.0091731*T+132.870)
     *T)))
2070 C5=FNA(.001964*SA):C6=FNA(
     .002541*SA):C8=FNA(-.02469
     1*SA)
2080 C9=FNA(-.004328*SIN(AN+FNA
     (275.05)-FNA(2.30*T)))
2090 LL=LL+C1+CT+C5
2100 MS=MS+C2
2110 ML=ML+C3+CT+C6
2120 DL=DL+C4+CT+C5
2130 FL=FL+CT+C8+C9
2140 :
2150 REM ** TERMINE CORRETTIVO
     .E. PER LONGITUDINE, LATIT
     UDINE E PARALLASSE LUNARE
     **
2160 :
2170 E1=1+(-.00000752*T-.002495
     )*T:E2=E1*E1
2180 :
2190 REM ** CALCOLO DELLA LONGI
     TUDINE VERA DELLA LUNA **
2200 :
2210 L1=6.288750*SIN(ML)+1.2740
     18*SIN(2*DL-ML)+.658309*SI
     N(2*DL)+.213616*SIN(2*ML)-
     .185596*SIN(MS)*E1-.114336
     *SIN(2*FL)+.058793*SIN(2*D
     L-2*ML)
2220 L1=L1+.057212*SIN(2*DL-MS-
     ML)*E1+.053320*SIN(2*DL+ML
     )+.045874*SIN(2*DL-MS)*E1+
     .041024*SIN(ML-MS)*E1-.034
     718*SIN(DL)-.030465*SIN(MS
     +ML)*E1
2230 L1=L1+.015326*SIN(2*DL-2*F
     L)-.012528*SIN(2*FL+ML)-.0
     10980*SIN(2*FL-ML)+.010674
     *SIN(4*DL-ML)+.010034*SIN(
     3*ML)+.008548*SIN(4*DL-2*M
     L)
2240 L1=L1-.007910*SIN(MS-ML+2*
     DL)*E1-.006783*SIN(2*DL+MS
     )*E1+.005162*SIN(ML-DL)+.0
     05000*SIN(MS+DL)*E1+.00404
     9*SIN(ML-MS+2*DL)*E1+.0039
     96*SIN(2*ML+2*DL)
2250 L1=L1+.003862*SIN(4*DL)+.0
     03665*SIN(2*DL-3*ML)+.0026
     95*SIN(2*ML-MS)*E1+.002602
     *SIN(ML-2*FL-2*DL)+.002396
     *SIN(2*DL-MS-2*ML)*E1-.002
     349*SIN(ML+DL)
2260 L1=L1+.002249*SIN(2*DL-2*M
     S)*E2-.002125*SIN(2*ML+MS)
     *E1-.002079*SIN(2*MS)*E2+.
     002059*SIN(2*DL-ML-2*MS)*E
     2-.001773*SIN(ML+2*DL-2*FL
     )-.001595*SIN(2*FL+2*DL)
2270 L1=L1+.001220*SIN(4*DL-MS-
     ML)*E1-.001110*SIN(2*ML+2*
     FL)+.000892*SIN(ML-3*DL)-.
     000811*SIN(MS+ML+2*DL)*E1+
     .000761*SIN(4*DL-MS-2*ML)*
     E1
2280 L1=L1+.000717*SIN(ML-2*MS)
     *E2+.000704*SIN(ML-2*MS-2*
     DL)*E2+.000693*SIN(MS-2*ML
     +2*DL)*E1+.000598*SIN(2*DL
     -MS-2*FL)*E1+.000550*SIN(M
     L+4*DL)
2290 L1=L1+.000538*SIN(4*ML)+.0
     00521*SIN(4*DL-MS)*E1+.000
     486*SIN(2*ML-DL)
2300 L1=FNA(L1):L2=LL+L1
2310 :
2320 REM ** CALCOLO DELLA LATIT
     UDINE DELLA LUNA **
2330 :
2340 BC=5.128189*SIN(FL)+.28060
     6*SIN(ML+FL)+.277693*SIN(M
     L-FL)+.173238*SIN(2*DL-FL)
     +.055413*SIN(2*DL+FL-ML)+.
     046272*SIN(2*DL-FL-ML)+.03
     2573*SIN(2*DL+FL)
2350 BC=BC+.017198*SIN(2*ML+FL)
     +.009267*SIN(2*DL+ML-FL)+.
     008823*SIN(2*ML-FL)+.00824
     7*SIN(2*DL-MS-FL)*E1+.0043
     23*SIN(2*DL-FL-2*ML)+.0042
     00*SIN(2*DL+FL+ML)
```

```
2360 BC=BC+.003372*SIN(FL-MS-2*
     DL)*E1+.002472*SIN(2*DL+FL
     -MS-ML)*E1+.002222*SIN(2*D
     L+FL-MS)*E1+.002072*SIN(2*
     DL-FL-MS-ML)*E1+.001877*SI
     N(FL-MS+ML)*E1
2370 BC=BC+.001828*SIN(4*DL-FL-
     ML)-.001803*SIN(FL+MS)*E1-
     .001750*SIN(3*FL)+.001570*
     SIN(ML-MS-FL)*E1-.001487*S
     IN(FL+DL)-.001481*SIN(FL+M
     S+ML)*E1
2380 BC=BC+.001417*SIN(FL-MS-ML
     )*E1+.001350*SIN(FL-MS)*E1
     +.001330*SIN(FL-DL)+.00110
     6*SIN(FL+3*ML)+.001020*SIN
     (4*DL-FL)+.000833*SIN(FL+4
     *DL-ML)
2390 BC=BC+.000781*SIN(ML-3*FL)
     +.000670*SIN(FL+4*DL-2*ML)
     +.000606*SIN(2*DL-3*FL)+.0
     00597*SIN(2*DL+2*ML-FL)+.0
     00492*SIN(2*DL+ML-MS-FL)*E
     1
2400 BC=BC+.000450*SIN(2*ML-FL-
     2*DL)+.000439*SIN(3*ML-FL)
     +.000423*SIN(FL+2*DL+2*ML)
     +.000422*SIN(2*DL-FL-3*ML)
     +.000367*SIN(MS+FL+2*DL-ML
     )*E1
2410 BC=BC-.000353*SIN(MS+FL+2*
     DL)*E1+.000331*SIN(FL+4*DL
     )+.000317*SIN(2*DL+FL-MS+M
     L)*E1+.000306*SIN(2*DL-2*M
     S-FL)*E2-.000283*SIN(ML+3*
     FL)
2420 W1=.0004664*COS(AN):W2=.00
     00754*COS(AN+FNA(275.05)-F
     NA(2.30*T))
2430 BT=FNA(BC*(1-W1-W2))
2440 :
2450 REM ** CALCOLO DELLA PARAL
     LASSE LUNARE **
2460 :
2470 PL=.950724+.051818*COS(ML)
     +.009531*COS(2*DL-ML)+.007
     843*COS(2*DL)+.002824*COS(
     2*ML)+.000857*COS(2*DL+ML)
     +.000533*COS(2*DL-MS)*E1
2480 PL=PL+.000401*COS(2*DL-MS-
     ML)*E1+.000320*COS(ML-MS)*
     E1-.000271*COS(DL)-.000264
     *COS(MS+ML)*E1-.000198*COS
     (2*FL-ML)+.000173*COS(3*ML
     )
2490 PL=PL+.000167*COS(4*DL-ML)
     -.000111*COS(MS)*E1+.00010
     3*COS(4*DL-2*ML)-.000084*C
     OS(2*ML-2*DL)-.000083*COS(
     2*DL+MS)*E1+.000079*COS(2*
     DL+2*ML)
2500 PL=PL+.000072*COS(4*DL)+.0
     00064*COS(2*DL-MS+ML)*E1-.
     000063*COS(2*DL+MS-ML)*E1+
     .000041*COS(MS+DL)*E1+.000
     035*COS(2*ML-MS)*E1-.00003
     3*COS(3*ML-2*DL)
2510 PL=PL-.000030*COS(ML+DL)-.
     000029*COS(2*FL-2*DL)-.000
     029*COS(2*ML+MS)*E1+.00002
     6*COS(2*DL-2*MS)*E2-.00002
     3*COS(2*FL-2*DL+ML)+.00001
     9*COS(4*DL-MS-ML)*E1
2520 :
2530 REM ** CALCOLO A.R. GEOCEN
     TRICA DELLA LUNA **
2540 :
2550 T1=SIN(L2)*COS(ET)-TAN(BT)
     *SIN(ET):T2=COS(L2)
2560 IF T2=0 THEN AL=π/2:GOTO 2
     610
2570 AL=ATN(T1/T2)-π*(T1>0 AND
     T2<0)-π*(T1<0 AND T2<0)-2*
     π*(T1<0 AND T2>0)
2580 :
2590 REM ** DECLINAZIONE GEOCEN
     TRICA DELLA LUNA **
2600 :
2610 LD=FNTA(SIN(BT)*COS(ET)+CO
     S(BT)*SIN(ET)*SIN(L2))
2620 :
2630 REM *** CORREZIONI IN A.R.
     E DECLINAZIONE DELLA LUNA
     PER NUTAZIONE E ABERRAZIO
     NE ***
2640 :
2650 DN=(COS(ET)+SIN(ET)*SIN(AL
     )*TAN(LD))*DF-COS(AL)*TAN(
     LD)*DE
2660 DD=SIN(ET)*COS(AL)*DF+SIN(
     AL)*DE
2670 D1=FNA(-20.49/3600)*(COS(A
     L)*COS(LV)*COS(ET)+SIN(AL)
     *SIN(LV))/COS(LD)
2680 D2=FNA(-20.49/3600)*(COS(L
     V)*COS(ET)*(TAN(ET)*COS(LD
     )-SIN(AL)*SIN(LD))+COS(AL)
     *SIN(LD)*SIN(LV))
2690 AL=AL+DN+D1:LD=LD+DD+D2
2700 :
2710 REM ** CALCOLO DEL TEMPO S
     IDERALE LOCALE **
2720 :
2730 JD=JJ-INT(JJ):IF JD=>.5 TH
     EN JD=INT(JJ)+.5:ELSE JD=I
     NT(JJ)-.5
2740 T=(JD-XN)/36525
2750 IU=0:TS=.276919398 +(100.0
     021359+.000001075*T)*T:TS=
     (TS-INT(TS))*24+HH*1.00273
     7908:TS=FNA(TS*15+LE)
2760 :
2770 REM ** TEMPO SIDERALE LOCA
     LE APPARENTE **
2780 :
2790 TS=FNC(TS+DF*COS(ET))
2800 :
2810 REM ** CALCOLO PARALLASSE
     E DI SIN(π) PER IL SOLE **
2820 :
2830 SP=8.794/R:S1=SIN(FNA(SP/3
     600))
2840 :
2850 REM ** CALCOLO DI SIN(π) P
     ER LA LUNA **
2860 :
2870 P1=SIN(FNA(PL))
2880 :
2890 IF AZ THEN AP=FNC(AR):AF=F
     NC(AL):PD=DC:FD=LD:GOTO 31
     30: REM ** PER ECLISSI DI
     LUNA **
2900 :
2910 REM ** CALCOLO RO*COS FI E
     RO*SIN FI **
2920 :
2930 U1=ATN(.99664719*TAN(LA))
2940 SF=.99664719*SIN(U1)+SIN(L
     A)*(HU/6378140)
2950 CF=COS(U1)+COS(LA)*(HU/637
     8140)
2960 :
2970 REM ** CORREZIONI IN A.R.
     E DECLINAZIONE DEL SOLE PE
     R LA SUA PARALLASSE ***
2980 :
2990 HA=TS-AR
3000 DA=ATN(-CF*S1*SIN(HA)/(COS
     (DC)-CF*S1*COS(HA)))
3010 AP=FNC(AR+DA)
3020 PD=ATN(((SIN(DC)-SF*S1)*CO
     S(DA))/(COS(DC)-CF*S1*COS(
     HA)))
3030 :
3040 REM ** CORREZIONI IN A.R.
     E DECLINAZIONE DELLA LUNA
     PER LA SUA PARALLASSE **
3050 :
3060 HL=TS-AL
3070 DA=ATN(-CF*P1*SIN(HL)/(COS
     (LD)-CF*P1*COS(HL)))
3080 AF=FNC(AL+DA)
3090 FD=ATN(((SIN(LD)-SF*P1)*CO
     S(DA))/(COS(LD)-CF*P1*COS(
     HL)))
3100 :
3110 REM ** CALCOLO DELLE COORD
     INATE DEL SOLE ALL'ORIZZON
     TE LOCALE **
3120 :
3130 HA=TS-AP
3140 O1=SIN(HA):O2=COS(HA)*SIN(
     LA)-TAN(PD)*COS(LA)
3150 IF O2=0 THEN ZS=π/2:GOTO 3
     210
3160 ZS=ATN(O1/O2)
3170 ZS=ZS-π*(O1>0 AND O2<0)-π*
     (O1<0 AND O2<0)-2*π*(O1<0
     AND O2>0)
3180 :
3190 REM ** AZIMUT DEL SOLE CON
     TATO DA NORD VERSO EST **
3200 :
3210 ZS=FNE(FNB(ZS)+180)
3220 :
3230 HZ=SIN(LA)*SIN(PD)+COS(LA)
     *COS(PD)*COS(HA):IF (1-HZ*
     HZ)=0 THEN HZ=SGN(HZ)*π/2:
     GOTO 3280
3240 HZ=FNTA(HZ)
3250 :
3260 REM ** ALTEZZA DEL SOLE SU
     LL'ORIZZONTE **
3270 :
3280 HZ=FNB(HZ)
3290 :
3300 REM ** CALCOLO DELLE COORD
     INATE DELLA LUNA ALL'ORIZZ
     ONTE LOCALE **
3310 :
3320 HL=TS-AF
3330 O1=SIN(HL):O2=COS(HL)*SIN(
     LA)-TAN(FD)*COS(LA)
3340 IFO2=0 THEN ZL=π/2:GOTO 34
     00
3350 ZL=ATN(O1/O2)
3360 ZL=ZL-π*(O1>0 AND O2<0)-π*
     (O1<0 AND O2<0)-2*π*(O1<0
     AND O2>0)
3370 :
3380 REM ** AZIMUT DELLA LUNA C
     ONTATO DA NORD VERSO EST *
     *
3390 :
3400 ZL=FNE(FNB(ZL)+180)
3410 :
3420 ZH=SIN(LA)*SIN(FD)+COS(LA)
     *COS(FD)*COS(HL):IF (1-ZH*
     ZH)=0 THEN ZH=SGN(ZH)*π/2:
     GOTO 3470
3430 ZH=FNTA(ZH)
3440 :
3450 REM ** ALTEZZA DELLA LUNA
     SULL'ORIZZONTE **
3460 :
3470 ZH=FNB(ZH)
3480 :
3490 REM ** RAGGIO APPARENTE DE
     LLA LUNA IN GRADI **
3500 :
3510 DP=1/P1
3520 R2=562.0492/(DP*36)
3530 :
3540 REM ** RAGGIO APPARENTE DE
     L SOLE IN GRADI **
3550 :
3560 R1=(959.63/R)/3600
3570 :
3580 REM ** CALCOLO DEL GRAFICO
     PER ECLISSI DI SOLE E DI
     LUNA **
3590 :
3600 DK=ZS-ZL:DH=HZ-ZH
3610 IF HZ<-.5 THEN CHAR,18,8,C
     HR$(15)+"INVISIBILE DA QUE
```

```
      STE COORDINATE A QUEST'ORA
      "+CHR$(143),1
3620 G1=190/(ABS(DK)-1.8*(AZ=0)
      -2.8*(AZ=1)):DK=DK*G1:DH=D
      H*G1:R2=R2*G1
3630 CHAR,8,6,"F = FINE  M
      = MENU      <RETURN> = C
      AMBIA L'ISTANTE E/O IL LUO
      GO",1:PRINT CHR$(7)
3640 SLOW:GRAPHIC1,1:BOX,41,0,3
      19,190:DRAW,41,95 TO 45,95
      :DRAW,180,0 TO 180,4:DRAW,
      316,95 TO 319,95:DRAW,180,
      187 TO 180,190
3650 HB=95.5/G1:AB=139.5/G1:J1=
      HZ+HB:J2=HZ-HB:J3=ZS-AB:J4
      =ZS+AB
3660 CHAR,3,14,"A",1:CHAR,3,15,
      "L",1:CHAR,3,16,"T",1:CHAR
      ,3,17,"E",1:CHAR,3,18,"Z",
      1:CHAR,3,19,"Z",1:CHAR,3,2
      0,"A",1:CHAR,12,24,"AZIMUT
      ",1
3670 IF J1>90 GOTO  3690
3680 CHAR,0,0,RIGHT$(STR$(INT(J
      1*10+.5)/10),5)
3690 IF J2<-90 GOTO 3710
3700 CHAR,0,23,RIGHT$(STR$(INT(
      J2*10+.5)/10),5)
3710 IF AB>180 GOTO 3760
3720 IF J3<0 THEN J3=360+J3
3730 IF J4>360 THEN J4=J4-360
3740 CHAR,5,24,RIGHT$(STR$(INT(
      J3*10+.5)/10),5)
3750 CHAR,35,24,RIGHT$(STR$(INT
      (J4*10+.5)/10),5)
3760 CHAR,0,12,RIGHT$(STR$(INT(
      HZ*10+.5)/10),5)
3770 CHAR,20,24,RIGHT$(STR$(INT
      (ZS*10+.5)/10),5)
3780 IF AZ GOTO 3940: REM ** AL
       GRAFICO ECLISSE DI LUNA *
      *
3790 :
3800 REM ** ECLISSE DI SOLE **
3810 R1=R1*G1
3820 :
3830 IF R1>R2 THEN R1=R1+1:R2=R
      2-1:ELSE IF R2>R1 THEN R2=
      R2+1:R1=R1-1
3840 CIRCLE,180,95,R1
3850 IF DH<-95 OR DK<-180 GOTO
      3870
3860 CIRCLE,180-DK,95+DH,R2
3870 POKE 208,0:GETKEY A$
3880 IF A$="M" THEN RUN 160
3890 IF A$="F" THEN 480
3900 FAST:GRAPHIC 0:GRAPHIC 5:G
      OTO 660
3910 :
3920 REM ** ECLISSI DI LUNA **
3930 :
3940 UP=1.02*(SP/3600+PL+R1):UO
      =1.02*(SP/3600+PL-R1)
3950 CIRCLE,180,95,UP*G1
3960 CIRCLE,180,95,UO*G1
3970 GOTO 3850
3980 :
3990 REM ** RICERCA DELLE ECLIS
      SI **
4000 :
4010 SCNCLR: CHAR ,19,0,"   RIC
      ERCA DI ECLISSI DI SOLE E
      DI LUNA   ",1
4020 CHAR ,19,1,"      (C) - MEN
      ICHELLI MARCO - FIESOLE
       ",1
4030 WINDOW 0,3,79,24
4040 CHAR,20,5,"ECLISSI DI SOLE
       O DI LUNA (S/L)  ? "
4050 GETKEY N$:IF N$="L" THEN N
      $="LUNA":AZ=1:ELSE N$="SOL
      E"
4060 PRINT N$
4070 :
4080 REM **********************
      *************************
      **********
4090 REM **** CALCOLO DI RICERC
      A PER ECLISSI DI SOLE O DI
       LUNA ****
4100 REM **********************
      *************************
      **********
4110 :
4120 CHAR,17,7,"INSERISCI UN AN
      NO FRA IL 1000 E IL 3000 >
      ":MN=47:MX=58:UZ=4:GOSUB 5
      20:A5=VAL(UZ$):IF A5<1000
      OR A5>3000 GOTO 4120
4130 T=(A5-1900)/100:DT=INT(.41
      +1.2053*T+.4992*T*T)
4140 SCNCLR:PRINT TAB(27)"*****
      * ECLISSI DI "N$" *****":I
      F DT<>0 THEN PRINT:PRINT T
      AB(27);"T.U. = T.E. - (";:
      PRINT USING "-## MINUTI )"
      ;DT
4150 PRINT CHR$(2);:CHAR,0,4,"
             TIPO
      GRANDEZZA       VISIBILITA'
           DATA     T.E.: H  M
      ":PRINT CHR$(130):CHAR,0,6
      ,""
4160 K=(A5-1900)*12.3685
4170 IF AZ=0 THEN K=INT(K)+1:GO
      TO 4230
4180 IF (K-INT(K))<.5 THEN K=IN
      T(K)+.5:GOTO 4230
4190 K=INT(K)+1.5
4200 :
4210 REM ** ISTANTE DELLE ECLIS
      SI **
4220 :
4230 T=K/1236.85
4240 :
4250 REM ** ANOMALIA MEDIA DEL
      SOLE ALL'ISTANTE .T. **
4260 :
4270 MS=359.2242+29.10535608*K+
      ((-.00000347*T-.0000333)*T
      )*T
4280 :
4290 REM ** ANOMALIA MEDIA DELL
      A LUNA ALL'ISTANTE .T. **
4300 :
4310 ML=306.0253+385.81691806*K
      +((.00001236*T+.0107306)*T
      )*T
4320 :
4330 REM ** ARGOMENTO DI LATITU
      DINE DELLA LUNA ALL'ISTANT
      E .T. **
4340 :
4350 F=21.2964+390.67050646*K+(
      (-.00000239*T-.0016528)*T)
      *T
4360 :
4370 F=FNA(FNE(F)):MS=FNA(FNE(M
      S)):ML=FNA(FNE(ML))
4380 :
4390 IF SIN(F)>.36 THEN K=K+1:G
      OTO 4230:REM ** ECLISSE IM
      POSSIBILE **
4400 :
4410 REM ** CALCOLO DEL GIORNO
      E DEL TEMPO (APPROSSIMATO)
       DELL'ECLISSE **
4420 :
4430 J=.75933+29.53058868*K+((-
      .000000155*T+.0001178)*T)*
      T
4440 J=J+.00033*SIN(FNA(166.56+
      (-.009173*T+132.87)*T))
4450 J=J+(.1734-.000393*T)*SIN(
      MS)+.0021*SIN(2*MS)-.4068*
      SIN(ML)+.0161*SIN(2*ML)
4460 J=J-.0051*SIN(MS+ML)-.0074
      *SIN(MS-ML)-.0104*SIN(2*F)
4470 J=J+.5
4480 :
4490 REM ** TRASFORMAZIONE DEL
      GIORNO GIULIANO IN GREGORI
      ANO **
4500 :
4510 I=INT(J)+XN:I1=J-INT(J):JJ
      =XN+(J-.5)
4520 IF I>2229160 THEN I2=INT((
      I-1867216.25)/36524.25):I3
      =I+1+I2-INT(I2/4):GOTO  45
      40
4530 I3=I
4540 I4=I3+1524:I5=INT((I4-122.
      1)/365.25):I6=INT(365.25*I
      5):I7=INT((I4-I6)/30.6001)
4550 I8=I4-I6+I1-INT(30.6001*I7
      ):I9=I7-1:IF I7>13.5 THEN
      I9=I7-13
4560 A1=I5-4716:IF I9<2.5 THEN
      A1=I5-4715
4570 HH=(I8-INT(I8))*24:H=INT(H
      H):MH=(HH-H)*60:MI=INT(MH)
      :I9=INT(I9)
4580 :
4590 IF A1<A5 THEN K=K+1:GOTO 4
      230:ELSE IF A1>A5 GOTO 507
      0
4600 :
4610 S=5.19595-.0048*COS(MS)+.0
      02*COS(2*MS)-.3283*COS(ML)
      -.006*COS(MS+ML)+.0041*COS
      (MS-ML)
4620 C=.2070*SIN(MS)+.0024*SIN(
      2*MS)-.0390*SIN(ML)+.0115*
      SIN(2*ML)-.0073*SIN(MS+ML)
4630 C=C-.0067*SIN(MS-ML)+.0117
      *SIN(2*F)
4640 :
4650 REM ** PER ECLISSI DI SOLE
       :
4660 REM ** DISTANZA FRA L'ASSE
       DELL'OMBRA LUNARE ED IL C
      ENTRO DELLA TERRA **
4670 REM ** OPPURE, PER ECLISSI
       DI LUNA :
4680 REM ** DISTANZA FRA IL CEN
      TRO DELLA LUNA E L'ASSE DE
      LL'OMBRA TERRESTRE **
4690 :
4700 L=S*SIN(F)+C*COS(F)
4710 :
4720 U=.0059+.0046*COS(MS)-.018
      2*COS(ML)+.0004*COS(2*ML)-
      .0005*COS(MS+ML)
4730 :
4740 IF AZ GOTO 4920: REM ** SA
      LTO PER ECLISSI DI LUNA **
4750 :
4760 IF ABS(L)>1.5432+U THEN K=
      K+1:GOTO 4230: REM ** L'EC
      LISSE DI SOLE NON AVVIENE
      **
4770 :
4780 IF ABS(L)=>.9972 AND ABS(L
      )<=1.5432+U THEN A$="PARZI
      ALE, TOTALE NON CENTR."
4790 M=(1.5432+U-ABS(L))/(.546+
      2*U)
4800 IF L>0 THEN B$="NORD"
4810 IF L<0 THEN B$="SUD"
4820 IF (ABS(L)=>.9972) AND (AB
      S(L)<=.9972+ABS(U))THEN A$
      ="PARZIALE, ANULARE NON CE
      NTR."
4830 IF L<.9972 AND L>-.9972 TH
      EN BEGIN:M=1
4840 IF U<0 THEN A$="TOTALE CEN
      TRALE"
4850 IF U>.0047 THEN A$="ANULAR
      E CENTRALE"
4860 IF U=>0 AND U<=.0047 THEN
      BEGIN
4870 W=FNTA(L):OM=.00464*COS(W)
4880 IF U<OM THEN A$="ANULARE T
      OTALE":ELSE A$="ANULARE"
```

```
4890 BEND
4900 BEND
4910 GOTO 5000
4920 B$="N/S":XC=2:M=(1.0129-U-
     ABS(L))/.545:A$="ECLISSE T
     OTALE":IF M<1 THEN XC=1:A$
     ="ECLISSE PARZIALE DI OMBR
     A"
4930 IF M<0 THEN BEGIN
4940 XC=1:M=(1.5572+U-ABS(L))/.
     545:A$="ECLISSE DI PENOMBR
     A"
4950 IF M<1 THEN A$="ECLISSE PA
     RZIALE DI PENOMBRA"
4960 BEND:IF M<.01 THEN K=K+1:G
     OTO 4230
4970 :
4980 REM ** COSTRUZIONE DELLA S
     TRINGA DI OUTPUT **
4990 :
5000 CC=CC+1:U(CC)=U:L(CC)=L:HH
     (CC)=HH:JJ(CC)=JJ:XC(CC)=X
     C:ML(CC)=ML:A$(CC)=A$:M(CC
     )=M:B$(CC)=B$:I8(CC)=INT(I
     8):I9(CC)=I9:H(CC)=H:MH(CC
     )=MH
5010 PRINT USING TP$;A$,M,B$,I8
     (CC),I9,A5,H,MH
5020 :
5030 REM ** RICERCA DI UN'ECLIS
     SE SUCCESSIVA **
5040 :
5050 K=K+1:GOTO 4230
5060 :
5070 IF CC=0 THEN CHAR,18,13,"N
     ON AVVENGONO ECLISSI DI LU
     NA NELL'ANNO"+STR$(A5),1:S
     LEEP 5:RUN 160
5080 :
5090 REM ** SCHERMATA PER LA SC
     ELTA DELL'ECLISSE DA OSSER
     VARE **
5100 :
5110 CHAR,22,13,"SCEGLI CON CRS
     R DOWN E PREMI RETURN",1
5120 CHAR,30,15,"F = FINE
     M = MENU"
5130 N=1
5140 POKE208,0:DOWHILEX$<>CHR$(
     13)
5150 CHAR,0,6
5160 FOR X=1 TO CC
5170 PRINT CHR$(18*-(N=X)-146*(
     N<>X));
5180 PRINT USING TP$;A$(X),M(X)
     ,B$(X),I8(X),I9(X),A5,H(X)
     ,MH(X)
5190 NEXT
5200 IF X$="F" OR X$="M" THEN E
     XIT
5210 GETKEY X$:IF X$=CHR$(17) T
     HEN N=N+1:IF N>CC THEN N=1
5220 LOOP
5230 :
5240 IF X$="F" THEN 480
5250 IF X$="M" THEN RUN 160
5260 :
5270 WINDOW 0,16,79,24:GRAPHIC
     5,1:U=U(N):L=L(N):HH=HH(N)
     :JJ=JJ(N):ML=ML(N)
5280 :
5290 IF XC=0 GOTO 640: REM ** E
     CLISSI DI SOLE **
5300 :
5310 REM ** CALCOLO DELLE CIRCO
     STANZE DELLE ECLISSI DI LU
     NA **
5320 :
5330 XC=XC(N):PZ=1.0129-U:NZ=.5
     458+.0400*COS(ML):IF PZ<AB
     S(L) THEN L=PZ:PZ=L(N)
5340 :
5350 IF XC=2 THEN BEGIN: REM **
      ECLISSE TOTALE DI LUNA **
5360 :
5370 TZ=.4679-U:IF TZ<ABS(L) TH
     EN L=TZ:TZ=L(N)
5380 TP=SQR(TZ*TZ-L*L)/NZ:H2=FN
     F(HH-TP):H3=FNF(HH+TP)
5390 BEND
5400 PP=SQR(PZ*PZ-L*L)/NZ:H1=FN
     F(HH-PP):H4=FNF(HH+PP)
5410 GOTO 640
```


ARRETRATI

COMPLETATE LA VOSTRA COLLEZIONE

Sono disponibili i seguenti numeri arretrati di COMMODORE GAZETTE
Numeri 1-2-3-4/86; 1-2-3-4-5-6-7/87; 1-2-3-4-5-6/88

Nome e cognome ....
Indirizzo ....
Città .... Cap ....
Desidero ricevere i seguenti numeri arretrati ....
Per un totale di lire (lit. 16.000 ognuno) ....
☐ Allego assegno bancario
☐ Allego fotocopia della ricevuta del vaglia postale

Firma ....

Ritagliare (si accettano anche fotocopie) e spedire a:
COMMODORE GAZETTE, Via Monte Napoleone, 9 - 20121 Milano

CORSO DI PROGRAMMAZIONE IN C PER L'AMIGA: SECONDA PUNTATA

PROGRAMMARE L'AMIGA CON IL SISTEMA MANX

Le caratteristiche di questo ambiente di programmazione vengono illustrate analizzando i file che ogni tool di programmazione richiede in input e genera in output

di Eugene P. Mortimore

Nel precedente articolo dedicato al linguaggio C abbiamo discusso le procedure generali per programmare in C sull'Amiga. Sono stati messi in evidenza i passi per compilare, assemblare e sottoporre alla fase di link un programma in C, ed è stato analizzato il flusso di controllo di un programma senza precisare il contenuto delle sue funzioni: tutte informazioni abbastanza generiche, valide per qualunque ambiente di programmazione. Ora discuteremo gli strumenti di lavoro del programmatore, gli indispensabili tool che permettono di trasformare un'idea in un programma eseguibile di buon livello.

I due sistemi di programmazione più popolari per l'Amiga sono il Lattice e il Manx. I rispettivi meriti dei due pacchetti sono stati discussi e ridiscussi sin da quando hanno fatto la loro apparizione sul mercato.

In questa puntata iniziamo un'analisi del sistema di programmazione *Aztec C68K V3.6* della Manx, uno dei più famosi in commercio, ma in futuro daremo spazio anche alla recentissima versione 5.0 del sistema Lattice, che presenta alcune novità molto interessanti.

A mano a mano che l'Amiga si è evoluto, la Manx ha dedicato molta attenzione ai vari miglioramenti e ha continuamente aggiornato e perfezionato il proprio ambiente di programmazione in linguaggio C. Ora il sistema Manx è cresciuto a tal punto da rappresentare uno dei migliori strumenti di programmazione sull'Amiga, se non il migliore. A testimonianza delle sue capacità, si consideri che molti dei più famosi programmi commerciali per l'Amiga sono stati creati con la versione Developer di questo pacchetto.

Il sistema Manx è molto flessibile: grazie agli strumenti che mette a disposizione si possono sviluppare sia piccoli programmi composti da un solo modulo, sia grandi programmi composti da più moduli gestiti in overlay. Il manuale che lo accompagna è abbastanza completo, e fornisce risposte dettagliate alla maggior parte dei problemi in cui può imbattersi il programmatore. Se dovesse risultare insufficiente, ci si può rivolgere direttamente al gruppo di supporto tecnico della Manx. Il manuale è composto da circa 300 pagine riunite in un raccoglitore ad anelli, pensato proprio per dare spazio agli aggiornamenti che vengono a mano a mano distribuiti. Gli argomenti sono organizzati in sezioni, da quelle introduttive a quelle che elencano nei dettagli le peculiarità del pacchetto.

Iniziamo parlando dell'organizzazione generale del sistema, e dei file che i suoi tool possono creare durante la realizzazione di un programma in C.

Il sistema di programmazione in C della Manx

L'ultima versione del compilatore *Aztec C68K* della Manx, la 3.6, è in grado di generare codici eseguibili per le più famose CPU Motorola: il 68000, il 68010, il 68020 nonché il coprocessore matematico 68881. La versione 3.6 genera inoltre un codice più veloce e compatto rispetto alle versioni precedenti, e le sue caratteristiche sono tanto articolate e numerose che questo articolo non basterebbe nemmeno a farne un elenco.

Questo pacchetto per l'Amiga è molto simile a quelli che la Manx ha creato per le macchine IBM, Atari, e per il Macintosh della Apple. Quindi gli aspetti fondamentali di questo ambiente di programmazione sono simili per tutte queste macchine: ciò che si impara sulla versione per l'Amiga può

facilmente essere utilizzato anche per gli altri computer.

Grazie al suo recente *Source Level Debugger*, il sistema Manx è ora considerato da molti il più importante ambiente di programmazione in C per l'Amiga: questo debugger permette di analizzare i codici sorgente in C o in Assembly eseguendo passo passo i codici macchina, e rendendo così molto rapido il passaggio al prodotto finale. Per queste e altre caratteristiche costituisce uno strumento di debug assolutamente indispensabile.

Il pacchetto della Manx comprende due dischi o tre a seconda che si tratti del sistema Professional o Developer. Con il secondo viene fornito un debugger simbolico, mentre il *Source Level Debugger* viene venduto solo separatamente e costituisce un pacchetto indipendente da entrambi i sistemi.

Il sistema Professional comprende tutti gli strumenti necessari per poter creare una vasta gamma di utili programmi in C. Contiene un compilatore, un assembler, un linker e un certo numero di librerie di tipo linked (si veda *Commodore Gazette* n. 6/88) che riuniscono numerose funzioni standard già compilate, insieme con una serie di file INCLUDE (alcuni dei quali specifici per l'Amiga) per la programmazione in C e in Assembly. Inoltre, il sistema Professional contiene diversi utili esempi di programmi sorgente in C che possono essere impiegati per prendere confidenza con il sistema Manx e con l'Amiga. Sebbene questa versione venga considerata ideale per sviluppare piccoli programmi e muovere i primi passi, è importante rendersi conto che non esiste alcun limite alla complessità dei programmi che permette di realizzare, anche se la mancanza di alcuni importanti strumenti rende il lavoro sempre più difficile a mano a mano che la complessità dei programmi aumenta. Inoltre, con questo sistema non viene fornito nessun editor di testi (cioè nessun programma per digitare i file sorgente, se non il solito ED previsto dall'AmigaDOS) e occorre quindi procurarsene uno per conto proprio.

Il sistema Developer è stato invece pensato per facilitare la produzione di complessi programmi composti da molti moduli. Ai file presenti nel sistema Professional, questo sistema aggiunge diverse altre utility che servono a rendere più efficiente il lavoro del programmatore, nonché speciali librerie di supporto matematico per la nuova release del software sistema dell'Amiga, la 1.2.1; si tratta di librerie contenenti funzioni matematiche per il coprocessore matematico 68881, e librerie atte a emulare le funzioni matematiche previste dallo standard IEEE (a queste librerie si fa ricorso quando è necessario eseguire qualche calcolo di una certa complessità senza possedere il coprocessore matematico).

Rispetto al sistema Professional, il sistema Developer aggiunge anche altri esempi di programmi sorgente in linguaggio C, che possono servire a migliorare la propria preparazione. Si tenga presente, infatti, che l'analisi di programmi funzionanti costituisce sempre un ottimo metodo diretto per imparare a programmare.

Il sistema *Aztec C68K* può essere acquistato direttamente alla Manx, il cui indirizzo è:

Manx Software Systems
P.O. Box 55
Shrewsbury, NJ 07701, USA
(tel. 001/201/542-2121)

I tool disponibili con il sistema Manx

I tool che il sistema Manx mette a disposizione del programmatore possono essere suddivisi in due categorie: quelli disponibili su entrambi i sistemi e quelli disponibili solo nel sistema Developer. La Tavola 1 (nella pagina successiva) elenca i più importanti differenziandoli nelle due categorie. Nel corso dell'articolo analizzeremo quelli che ricorrono più di frequente nella programmazione.

I primi tre tool elencati nella tavola, *cc*, *as* e *ln*, sono rispettivamente il compilatore, l'assembler e il linker. Questi sono i principali strumenti di lavoro del sistema. Il tool *mclk* viene mandato automaticamente in esecuzione durante la startup-sequence. Si tratta di una semplicissima finestra, composta solo dalla barra di spostamento, nella quale oltre all'ora di sistema appaiono le indicazioni dei kilobyte di chip RAM e fast RAM disponibili. Queste due informazioni vengono aggiornate in tempo reale, permettendo al programmatore di verificare in ogni momento la disponibilità della RAM di sistema. Se la presenza di questa finestra sullo schermo dovesse infastidire, basta chiuderla con il mouse. È interessante notare che di questa applicazione viene anche fornito il sorgente.

Il tool *set* viene usato per impostare, ed eventualmente rivedere, lo stato di particolari variabili relative all'ambiente di programmazione; queste variabili, dette variabili di sistema, in genere vengono assegnate per la prima volta nel file comandi s/startup-sequence che l'Amiga esegue durante il boot (si ricordi che un file comandi è un testo contenente comandi che vengono eseguiti in sequenza, come se fossero impartiti dalla tastiera). È possibile utilizzare le variabili di sistema per specificare al sistema Manx in quale directory si trovano la maggior parte dei file coinvolti nelle procedure di programmazione, cosicché ogni tool sappia come accedere a quelli di cui ha bisogno. Queste particolari variabili configurano l'intero ambiente di lavoro. Per ora ci basta sapere che esistono e che sono importanti; nel prossimo articolo il loro impiego verrà illustrato dettagliatamente da un esempio.

L'ultimo tool degno di nota (che compare in entrambe le versioni) è *lb*. Questa utility serve a creare e gestire le librerie linked: il programmatore può usarla per costruire librerie linked personalizzate che riuniscano nuove funzioni, scritte in C o in Assembly, compilate e verificate, da affiancare a quelle contenute nelle librerie linked del linguaggio C standard. La necessità di crearsi una libreria linked personale può nascere quando, realizzando

Nome del programma	Posizione del programma	Descrizione del programma
File che si trovano in entrambi i sistemi		
cc	Disco1	Compilatore C
as	Disco1	Assembler
ln	Disco1	Linker
mclk	Disco1	Orologio e indicatore della memoria disponibile
set	Disco1	Utility per definire variabili di sistema
setdat	Disco1	Utility per impostare la data di sistema
lb	Disco2	Utility per gestire i file oggetto delle librerie
File che si trovano solo nel sistema Developer		
adump	Disco3	Utility per visualizzare informazioni su un particolare file eseguibile
avail	Disco3	Rileva la quantità di memoria disponibile
acvt	Disco3	Utility per convertire moduli oggetto in moduli sorgente Assembly
arcv	Disco3	Utility per archiviare un file sorgente in C
cat	Disco3	Programma per visualizzare il contenuto di più file
cmp	Disco3	Programma generico per il confronto carattere per carattere dei file
cnm	Disco3	Riporta informazioni sui file oggetto
ctags	Disco3	Viene usato per creare i file tags per l'editor Z
db	Disco3	Debugger simbolico – NON è il Source Level Debugger
diff	Disco3	Programma di derivazione UNIX per confrontare file sorgente
du	Disco3	Utility per migliorare la gestione delle directory: visualizza i blocchi usati dalla directory
grep	Disco3	Programma di derivazione UNIX per la ricerca delle stringhe nei file
hd	Disco3	Utility per la visualizzazione esadecimale e ASCII dei file
ls	Disco3	Programma per la visualizzazione e l'ordinamento dei file
make	Disco3	Programma di derivazione UNIX per la gestione di più file sorgente
mkarcv	Disco3	Programma per dearchiviare i file sorgente (si veda arcv)
obd	Disco3	Programma per ottenere l'elenco dei simboli nei file oggetto
ord	Disco3	Ordina gli elenchi ottenuti con il comando precedente – utile per costruire librerie linked
touch	Disco3	Utility per aggiornare la data e l'ora di creazione dei file
Z	Disco3	Editor di file sorgente a finestra singola, di derivazione UNIX. Ha la possibilità di gestire file tags

Tavola 1: sommario dei programmi eseguibili e delle utility presenti nel sistema di programmazione Aztec C68K della Manx

programmi di una certa estensione, vi sono funzioni non standard a cui si ricorre con frequenza (e che dovrebbero essere compilate con ogni modulo).

La seconda categoria di tool elencati nella Tavola 1 è disponibile soltanto nel sistema Developer, e raccoglie quelli che si trovano sul disco 3 della confezione distribuita dalla Manx. Fra questi tool notiamo l'editor *Z* e il debugger simbolico *db*. A questo proposito, è bene ribadire che il debugger simbolico *non è* il *Source Level Debugger*, ma uno strumento senza dubbio meno versatile e anche più difficile da usare.

Oltre a questi due tool è importante citare *grep*, uno strumento di ricerca tipico dell'ambiente UNIX che permette di cercare una particolare stringa di caratteri all'interno di più file. Spesso *grep* è utile per rintracciare dati nei programmi sorgente scritti in C, quando si cerca per esempio il nome di una struttura nei file INCLUDE. È interessante notare che questo tool, a differenza di quanto avviene con l'AmigaDOS, prevede caratteri jolly di ricerca molto semplici e consente di effettuare ricerche estremamente articolate. Per avere altri dettagli consigliamo di riferirsi alla documentazione fornita dalla Manx.

La gestione dei file nel sistema Manx

L'ambiente di programmazione della Manx è un complesso sistema di elaborazione delle informazioni; nella configurazione base, si articola in cinque moduli separati ma interdipendenti, detti anche tool di programmazione. Per quanto il manuale della Manx sia ben fatto, può capitare di non capire appieno l'organizzazione dei file creati e gestiti nel sistema. Per questa ragione è di grande aiuto esaminare tutti i file coinvolti, ossia capire come vengono scritti dai singoli strumenti di programmazione e che uso ne viene fatto. L'obiettivo finale di questo studio è personalizzare le varie fasi del lavoro e automatizzare tramite opportuni file comandi la maggior parte delle procedure che si seguirebbero di volta in volta.

A tal fine esaminiamo la Figura 1. Per semplicità, ipotizziamo di voler realizzare un programma in C "puro", cioè che non contenga nel sorgente codici Assembly. Tutti i file richiesti dal compilatore e dal linker possono essere indicati una volta per tutte all'interno di un file comandi, in modo da rendere completamente automatica la trasformazione da file sorgente a file eseguibile. La Figura 1

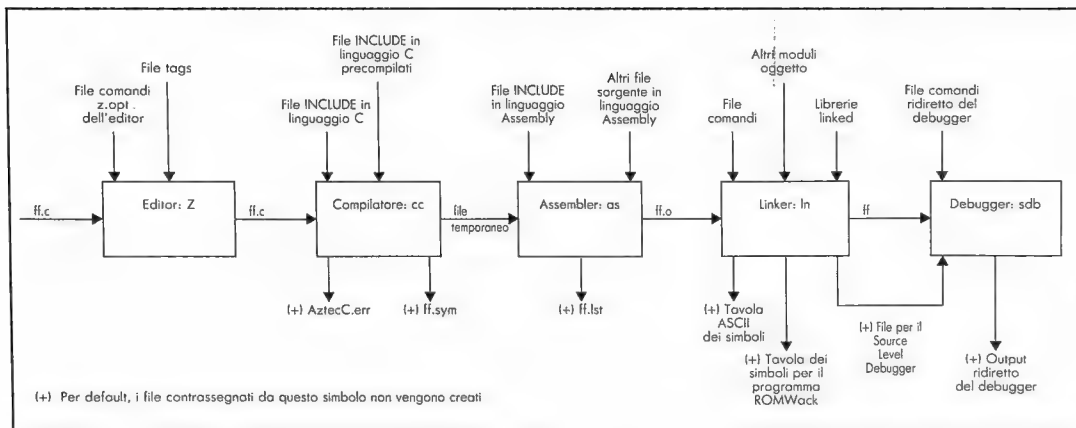



Figura 1: organigramma dei file contenuti nel pacchetto di programmazione in C della Manx, in ordine di esecuzione

mostra la provenienza, la destinazione e il significato di tutti i file che intervengono nella programmazione. Nella figura appaiono cinque rettangoli che rappresentano ciascuno un particolare strumento di programmazione, come l'editor e il compilatore; sono i tool che leggono e generano i diversi file del sistema. Iniziando da sinistra, abbiamo l'editor *Z*, il compilatore *cc*, l'assembler *as*, il linker *ln*, e alla fine il *Source Level Debugger sdb*.

Nella programmazione, l'editor e il debugger possono essere considerati come accessori, mentre il compilatore, l'assembler e il linker sono tool indispensabili per sviluppare programmi in C. Questo significa che è possibile usare qualsiasi editor di testi ASCII per produrre i file sorgente, e che non è necessario possedere il *Source Level Debugger* per sviluppare i propri programmi. Tuttavia si scopre ben presto che quest'ultimo tool risulta molto utile per identificare i bug meno evidenti che si possono annidare nei programmi. A questo proposito ricordiamo che nella creazione di un programma, molti problemi si possono eliminare già nella fase di stesura, prima di procedere alla compilazione. Si tratta degli errori di sintassi che a volte vengono inseriti nel sorgente solo per stanchezza o per disattenzione. Per identificarli si può ricorrere a un tool di controllo sintattico del sorgente, come il *Source File Syntax Checker Gimple Lint*. È possibile acquistare questo tool sia direttamente dalla Manx sia attraverso il proprio rivenditore.

Per accelerare i tempi tutti questi tool di programmazione, o almeno il linker e il compilatore, andrebbero copiati ogni volta nel dispositivo logico RAM: (il cosiddetto RAM disk). Questa operazione si può rendere automatica inserendo nella startup-sequence il comando COPY bin TO RAM:, tanto più che fra i vari percorsi di ricerca (path) che il sistema prende in considerazione, il dispositivo logico RAM: è il primo (per verificarlo, basta semplicemente impartire il comando PATH senza specificare nessun argomento).

Alcuni dei file di supporto che compaiono nella figura vengono scritti automaticamente dai tool di programmazione, mentre per altri è necessario l'intervento diretto del programmatore, che indica ai tool come operare facendo uso delle loro opzioni. I dettagli relativi alla gestione di tutti questi file diventeranno più chiari nel prossimo articolo, quando analizzeremo i file comandi di startup, di compilazione e di link.

Gli altri file non eseguibili presentati nella Figura 1 possono essere classificati secondo diversi criteri: il più ovvio distingue fra file di input e file di output di ogni tool. Nella figura i file di input sono quelli che appaiono sopra i tool di programmazione; si tratta di file, alcuni di testo e altri in formato non-ASCII, ai quali i tool accedono per ottenere informazioni. I file di output sono invece quelli prodotti dai tool di programmazione, e appaiono nella figura sotto questi ultimi. Si noti che spesso il file di output di un tool diviene il file di input del successivo, e che tra i file di output compaiono anche quelli prodotti dall'AmigaDOS quando si ridirige l'output dallo schermo verso un altro dispositivo logico (per esempio, un file su disco) tramite il simbolo di ridirezione ">"; questi particolari file di output non sono mostrati nella Figura 1.

I file previsti dall'editor di testi Z

Il primo rettangolo sul lato sinistro della Figura 1 rappresenta l'editor di testi Z della Manx. In generale, possiamo identificare quattro file legati a questo editor. Il primo è il file sorgente da creare o da modificare, che nella figura indichiamo con il nome ff.c. Si tratta di un file di testo contenente le istruzioni in C atte a definire un modulo di programma. Nella figura si vede che, tramite

l'editor, il programmatore accede al file ff.c (o lo crea), lo modifica (o lo scrive per la prima volta) e lo salva sul disco senza cambiarne il nome. Ovviamente, anche se la figura non lo mostra, questa prima fase può essere ripetuta molte volte prima che il file sorgente sia pronto per la compilazione, e continua fino a quando il programma non risulta completamente privo di errori.

Quando l'editor *Z* viene mandato in esecuzione, cerca automaticamente un file di startup convenzionalmente chiamato z.opt (se non lo trova prosegue ignorando questa fase). Questo file consente al programmatore di configurare l'editor secondo le proprie esigenze (tabulatori, word wrap, file di backup...). Normalmente la posizione del file z.opt nelle directory del disco viene indicata nella startup-sequence tramite la variabile di sistema ZOPT.

L'editor *Z* prevede in input un altro tipo di file, il cosiddetto "file tags", ovvero una tavola di riferimento che associa a determinati nomi di funzioni i nomi dei file sorgente nei quali queste funzioni sono definite. Se il file tags, come normalmente avviene, è stato creato tenendo conto di tutti i moduli sorgente che costituiscono il programma, permette all'editor *Z* di sapere esattamente in quale modulo sorgente è stata definita ogni funzione dell'intero programma. Questa caratteristica può essere utile per esempio quando il programmatore decide di controllare con l'editor il corpo principale del programma, e incontra una chiamata a una funzione di cui non ricorda il comportamento. È probabile che non ricordi nemmeno in quale modulo sorgente la funzione è stata definita, ma grazie al file tags basta indicarne il nome con un apposito comando e *Z* carica e visualizza il modulo sorgente nel quale la funzione è definita. In questo modo, e senza uscire dall'editor, il programmatore può facilmente muoversi tra i singoli moduli sorgente alla ricerca di una particolare funzione, anche se non ricorda dove si trova.

Il file tags che permette questa facile ricerca dev'essere opportunamente creato dal programmatore mandando in esecuzione l'utility *ctags* (si veda la Tavola 1) e indicando come argomenti tutti i moduli sorgente che devono essere presi in considerazione.

Per conoscere la posizione del file tags, l'editor *Z*, quando viene mandato in esecuzione, controlla se la variabile di sistema TAGS è stata inizializzata con il nome del file tags da considerare (compreso, ovviamente, il relativo percorso di ricerca). Il programmatore può inizializzare questa variabile direttamente o tramite la startup-sequence. Se TAGS non risulta inizializzata, oppure se il file tags indicato non è presente sul disco, l'editor *Z* controlla se esiste un file di nome tags nella directory selezionata in quel momento. Di fronte a un nuovo insuccesso, *Z* non compie altri tentativi e non rende disponibile la ricerca veloce delle funzioni fra i moduli sorgente.

Con la variabile di sistema TAGS, il programmatore in genere individua sul disco un file tags che vale per l'intero sistema di programmazione. Nel quale cioè sono indicati tutti i moduli sorgente abitualmente impiegati come supporto alla creazione dei programmi. Con un file tags di questo tipo, durante la stesura di un programma è possibile richiamare in ogni momento i moduli sorgente della propria biblioteca di funzioni preconfezionate.

Se però si desidera disporre anche di un particolare file tags meno generale e riferito soltanto ai moduli sorgente di un particolare programma, è sufficiente crearlo e indicarlo all'editor *Z* in sostituzione del file tags di sistema. Il disco 3 del sistema Developer contiene un esempio di file tags.

I file di input del compilatore C

Il compilatore Aztec C della Manx usa al massimo tre file di input. Proprio come nell'editor, alcuni di questi file sono necessari mentre altri sono opzionali. Il primo file di input è il sorgente stesso, ff.c, cioè il file che il compilatore trasformerà, chiamando anche l'assembler, in file oggetto. Se al suo interno vengono richiamati uno o più file INCLUDE, il compilatore accede anche a questi file, che rappresentano il secondo tipo di file di input del compilatore; la posizione di questi file nelle directory può essere specificata attraverso la variabile di sistema INCLUDE, oppure tramite l'opzione -I quando si manda in esecuzione il compilatore.

In ambedue i modi si segnala al compilatore la posizione dei file INCLUDE, tenendo presente però che i file INCLUDE specificati attraverso la variabile di sistema vengono considerati primari, mentre quelli specificati attraverso l'opzione -I del compilatore vengono considerati secondari. La divisione dei file INCLUDE in primari e secondari è sotto il controllo diretto del programmatore e dipende dal tipo d'informazioni di cui necessita il programma.

Ogni programma sorgente scritto in C contiene una serie di comandi #include che richiamano i file INCLUDE necessari per compilarlo correttamente. Oltre ai consueti file INCLUDE, il compilatore Manx accetta anche file INCLUDE precompilati. Questi rappresentano il terzo tipo di file di input: sono più piccoli dei file INCLUDE normali, permettono di risparmiare spazio sul disco, accelerano il processo di compilazione e sono mattoni importanti dei grandi progetti di programmazione.

Per indicare al compilatore che deve leggere file INCLUDE precompilati, si attiva l'opzione +I. Si veda la discussione dell'opzione +H più avanti e la Tavola 2 (a pagina 86) per ulteriori informazioni sui file INCLUDE precompilati.

I file di output del compilatore C

Il compilatore C della Manx produce al massimo quattro file di output, se non si tiene conto dei file temporanei. Il primo file di output è il file risultante dalla fase di "traduzione" in linguaggio Assembly sorgente del programma sorgente in C. Questo file è temporaneo e serve solo all'assembler quando il compilatore lo manda automaticamente in esecuzione. L'assembler lo riceve in input, crea il relativo file oggetto, e lo cancella dal disco. Questa è la situazione che si presenta quando ci si affida al comportamento di default del compilatore. È possibile definire nella variabile di sistema CCTEMP in quale directory dev'essere salvato il file temporaneo e dove, di conseguenza, l'assembler dovrà cercarlo quando riceve il controllo. In genere si dedica una subdirectory appositamente a questo scopo. In questa procedura il semplice risultato del compilatore (il file sorgente in Assembly) non è accessibile al programmatore.

Tramite opportune opzioni, specificate all'atto della chiamata, è possibile modificare il comportamento del compilatore. L'opzione -A consente di fermare il compilatore subito dopo la creazione del file temporaneo, consentendo così al programmatore l'analisi dei codici Assembly prodotti. Questo file mantiene lo stesso nome del file originale, seguito però dall'estensione ".asm", ed è un file di testo. L'analisi dei codici Assembly consente di verificare l'opera di traduzione svolta dal compilatore. A volte si scopre che certi codici si possono migliorare, oppure che il risultato ottenuto non è conforme a quello richiesto. Oppure, si possono analizzare i sorgenti in Assembly al fine di imparare questo linguaggio di basso livello partendo dalla conoscenza del C. A questo scopo, si può selezionare l'opzione -T del compilatore per vedere le istruzioni originali del programma sorgente in C intercalate con quelle in Assembly ottenute dopo la compilazione. Il Listato 2 (a pagina 87) è un esempio di file sorgente in Assembly; è stato ottenuto compilando il Listato 1. Si noti al suo interno la presenza delle istruzioni in C.

Se si procede in questo modo, l'assembler dev'essere poi mandato in esecuzione manualmente, il che consente di attivare opzioni che altrimenti non sarebbe possibile specificare nella linea comando del compilatore.

L'opzione -O consente di attribuire un nuovo nome al file che si ottiene al termine della compilazione. Normalmente si tratta del file oggetto, ma se questa opzione viene usata congiuntamente all'opzione -A, serve a indicare un nome nuovo per il file sorgente in Assembly.

Il secondo file di output è il file denominato AztecC.err. Il compilatore crea questo file se si specifica l'opzione -Q sulla sua linea comandi. Il file AztecC.err contiene un elenco degli errori rilevati durante la compilazione, esposti in un formato più dettagliato del normale output su video generato durante la compilazione. Se si prevede un discreto numero di errori, può essere utile scegliere questa opzione per analizzarli in maniera più approfondita.

Il terzo file di output viene prodotto dal compilatore se si specifica l'opzione +H seguita dal nome che si vuole attribuirgli; si tratta della cosiddetta "tavola dei simboli". Oltre a fornire informazioni sui simboli, ossia sulle variabili e sulle label del programma, che possono essere studiate tramite questo file, l'opzione +H si può usare per generare le tavole di simboli relative ad alcuni file INCLUDE, trasformandoli nei cosiddetti file INCLUDE precompilati. Nel nostro caso la tavola viene salvata nel file di nome ff.sym.

Oltre ai precedenti file, è possibile ridirigere l'output del compilatore su un ulteriore file tramite il comando ">" dell'AmigaDOS: questo file conterrà informazioni che normalmente verrebbero scritte sullo schermo dell'Amiga. Qui, coerentemente con le convenzioni sulla denominazione dei file usate altrove, chiameremo ff.e questo file, dove "e" significa "errori". Quando si ridirige l'output, si tenga presente che il compilatore interrompe automaticamente la compilazione dopo ogni 5 errori per chiedere al programmatore se deve continuare. Se l'output è stato ridiretto su un file, è immediato capire che questa interruzione non verrebbe mai rilevata. Per ovviare a questo inconveniente, è sufficiente indicare l'opzione -B, che ordina al compilatore di non bloccarsi fino a quando non completa il proprio lavoro, prescindendo dal numero di errori che si verificano.

Se infine si desidera sottoporre il programma al *Source Level Debugger*, in fase di compilazione si deve specificare l'opzione -N. In questo modo il compilatore crea anche un particolare file temporaneo che il linker successivamente impiega per generare il relativo file di supporto del *Source Level Debugger*.

I file di input dell'assembler

La Figura 1 mostra che l'assembler della Manx richiede al massimo tre file di input: il primo è il file sorgente temporaneo in linguaggio Assembly creato dal compilatore. L'intera operazione di scrittura e lettura relativa a questo file viene svolta automaticamente dal sistema Manx senza alcun intervento da parte del programmatore oltre all'assegnazione della variabile CCTEMP.

La categoria successiva di file dati in input all'assembler è costituita dai file INCLUDE in linguaggio Assembly. Questi vengono presi in considerazione soltanto nel caso che si programmi direttamente in Assembly, senza passare dal C. Si tenga infatti presente che l'assembler della Manx può essere utilizzato con profitto anche quando si devono sviluppare programmi in Assembly. Questi file sono contenuti nel disco 2 del pacchetto, all'interno della directory asm, e sono dotati dell'estensione ".i". Perché l'assembler li prenda in considerazione, devono essere richiamati esplicita-

mente all'interno dei programmi in linguaggio Assembly per mezzo di comandi include.

I file di output dell'assembler

L'assembler della Manx può produrre in tutto tre file di output: il primo è il programma oggetto, che nel nostro esempio prende il nome di ff.o (si noti l'estensione ".o" dei file oggetto). Un programma oggetto contiene una versione non ancora eseguibile del programma, ma già trasformata nei codici in linguaggio macchina del 68000: a questo stadio non sono ancora stati risolti tutti i riferimenti alle routine dichiarate in altri moduli, a locazioni di memoria del sistema, e alle librerie dell'Amiga. A meno che non si sia scelta l'opzione -O dell'assembler, il programma oggetto viene salvato automaticamente nella stessa directory in cui si trova il programma sorgente.

Il secondo file di output generato dall'assembler è il cosiddetto file list, il cui nome è ancora una volta quello del programma sorgente seguito dall'estensione ".lst". Questo file viene creato esclusivamente se è stata specificata l'opzione -L dell'assembler. In questo file di testo vengono mostrate le istruzioni in Assembly del programma e i loro corrispondenti codici in linguaggio macchina; vengono riportati anche gli indirizzi relativi (offset) di ciascuna istruzione del programma. Analizzando questo file di testo ci si può rendere conto di quali riferimenti assoluti debbano essere ancora risolti al termine della compilazione (compito che spetterà al linker). Il Listato 3 mostra il file list ottenuto assemblando il sorgente Assembly riportato nel Listato 2.

Il terzo file che l'assembler può produrre è il file derivante dalla ridirezione dell'output normalmente diretto allo schermo, in maniera analoga a quanto visto per l'assembler. In una tipica sequenza di comandi di compilazione e link non sarebbe possibile produrre questo file, perché l'assembler non viene mandato in esecuzione esplicitamente. Tuttavia nel prossimo articolo vedremo come ottenerlo ugualmente.

I file di input del linker

Questo programma accetta in input quattro file. Il primo è il modulo oggetto, ff.o, che proviene dall'assembler e che è stato salvato sul disco nella stessa directory del programma sorgente ff.c. Va messo in evidenza che, a differenza dell'assembler, il linker non viene mandato in esecuzione automaticamente e quindi richiede negli argomenti della chiamata i nomi dei file che deve elaborare.

I successivi file sono altri moduli oggetto direttamente correlati al modulo che si sta compilando: insieme con il blocco principale (quello che contiene la funzione main()) costituiscono l'intero programma. Per esempio, se il programma è composto da due file sorgente f1.c e f2.c, occorre dapprima produrre un modulo oggetto (f1.o) compilando e assemblando il primo file sorgente. Poi si sottopone alla stessa procedura f2.c, e si ottiene f2.o. A questo punto abbiamo due moduli oggetto dotati di reciproci riferimenti incrociati che devono essere risolti. Non rimane che sottoporli entrambi alla fase di link ottenendo infine il programma eseguibile ff (si noti che per default il file eseguibile creato dal linker è privo di estensione).

Analogamente è possibile richiamare durante la fase di link altri file oggetto specificandone i nomi nella linea comandi del linker. In particolare sarà necessario richiamare le librerie linked (come la libreria standard del C, c.lib). Va ricordato che queste librerie predefinite contengono le definizioni delle funzioni che vengono chiamate, ma non definite, nell'ambito del programma (per esempio le funzioni standard printf() e scanf()), ed è il linker che ne inserisce i relativi codici in linguaggio macchina nel programma eseguibile.

L'ultimo file di input del linker è il cosiddetto "command arguments file". È possibile ordinare al linker di leggere questo file attivando l'opzione -F. L'utilità di questo file risiede nella possibilità per il programmatore di specificare un insieme preconfezionato di opzioni che normalmente andrebbero indicate nella chiamata al linker. Si noti che questo è soltanto un modo diverso di dire al linker come deve trattare i suoi dati, cioè un altro modo per specificare le opzioni coinvolte. Tuttavia, se si usa un file comandi per le fasi di compilazione e di link, non ci sarà quasi mai bisogno di usare il command arguments file, specialmente per i programmi più semplici che si scrivono durante le prime fasi di apprendimento.

I file di output del linker

Il linker può produrre fino a cinque file. Il primo è il file eseguibile, ossia il prodotto finale del sistema Manx, corrispondente a uno o più moduli sorgente in C. A meno che non venga diversamente specificato tramite l'opzione -O, il linker assegna automaticamente al file eseguibile lo stesso nome del primo modulo oggetto indicato in input, privandolo però dell'estensione ".o". Questo file è il programma finale eseguibile dall'Amiga (si noti che la parola "eseguibile" non garantisce la perfetta esecuzione del programma).

Il secondo file di output è una tavola di simboli ASCII che viene prodotta automaticamente se si specifica l'opzione -T del linker e contiene una lista di tutti i simboli utilizzati dal programma: nomi di variabili e di locazioni di memoria, che possono essere usati per vedere dov'è allocata la memoria per ciascuna variabile e per ciascuna istruzione del programma. Un indirizzo, in questa tavola, può essere il punto d'ingresso di una funzione oppure la locazione di memoria relativa a una label del programma.

Il terzo file è anch'esso opzionale: si tratta di una tavola di simboli per il debugger residente

ROMWack dell'Amiga. Se si desidera ottenere questo file bisogna specificare l'opzione -W nella linea comandi del linker. Questo file servirà naturalmente solo se si prevede di usare il debugger dell'Amiga. Nel corso di questa serie supporremo che venga usato il *Source Level Debugger* della Manx, ma se non si dispone di questo programma, e non si possiede nemmeno il debugger simbolico del sistema Developer, è possibile chiamare questo file e utilizzare il debugger *ROMWack*. Si veda l'*Amiga ROM Kernel Reference Manual* per una spiegazione dei comandi e delle funzioni di questo debugger.

Il quarto file di output del linker è il file opzionale dedicato al *Source Level Debugger* della Manx. Il linker crea questo file, nel nostro esempio ff.dbg, se si specifica l'opzione -G (non è un file di testo). Si tenga presente che si può generare questo file soltanto se i moduli del programma sono stati compilati attivando l'opzione -N. Il *Source Level Debugger* della Manx ricerca automaticamente questo file di estensione ".dbg" quando si appresta a compiere il debug del file eseguibile ff.

L'ultimo file di output è ancora il file ottenuto dalla ridirezione dell'output prodotto dal linker.

I file di input del Source Level Debugger

Il rettangolo più a destra nella Figura 1 rappresenta l'ultimo tool del sistema Manx, il *Source Level Debugger*. Tramite questo tool di debug è possibile mandare in esecuzione un programma e contemporaneamente osservare le istruzioni in C o in Assembly del sorgente a mano a mano che vengono eseguite. Questo programma prevede al massimo tre file di input. Il primo è il programma eseguibile prodotto dal linker, sul cui corretto funzionamento è bene non scommettere, e che verrà quindi sottoposto al debug per individuare eventuali errori.

Il file di input successivo richiesto dal *Source Level Debugger* è il file che il linker associa al programma eseguibile se viene indicata l'opzione -G. Il *Source Level Debugger* utilizza questo file internamente per riordinare le informazioni presenti nel programma eseguibile a mano a mano che il processo di debug procede, e si aspetta che il suo nome sia dotato dell'estensione ".dbg" (nel nostro caso questo file si chiama ff.dbg).

A differenza dei precedenti, il terzo file di input è opzionale: si tratta del cosiddetto file comandi ridiretti, e rappresenta un set di comandi predefiniti che sono stati precedentemente studiati e salvati sul disco in un apposito file. Il *Source Level Debugger* lo prende in considerazione automaticamente se è stata indicata l'opzione -M, ma permette comunque di mandarlo in esecuzione durante qualsiasi fase del debug tramite un opportuno comando. Questo file è di enorme utilità una volta che si è acquisita una certa familiarità con il debugger e ci si accorge che durante il debug di un programma alcune sequenze di comandi sono ripetitive. Questa situazione si presenta di solito soprattutto quando si intraprende un progetto particolarmente ambizioso e la fase di debug continua molto a lungo.

I file di output del Source Level Debugger

Il *Source Level Debugger* produce un solo file di output, il file di output ridiretto. Non si tratta del file di ridirezione dell'output gestito dall'AmigaDOS, ma di un file creato dal tool stesso tramite particolari comandi; può avere due forme diverse. Nella prima consiste dei soli comandi impartiti durante il processo di debug; il file che si ottiene, eventualmente modificato, può essere utilizzato successivamente come file comandi ridiretti per il debugger stesso.

Nella seconda forma, il file di output ridiretto consiste non solo dei comandi impartiti, ma anche di tutti i dati ottenuti in output dal debugger stesso, cioè delle informazioni che normalmente vengono inviate soltanto allo schermo. Il programmatore, o il file comandi ridiretto, può attivare la creazione di questi due file in qualsiasi fase della procedura di debug. Questo file si presta particolarmente a un uso didattico: cosigliamo di esaminarlo alla fine della procedura di debug tramite l'editor oppure tramite lo stampato.

Elenco dei file previsti dal sistema Manx

La Tavola 2 (nella pagina successiva) riassume i punti più importanti di tutti i file presentati nella Figura 1. Esaminandola, si può rilevare se un dato file è sotto il controllo del programmatore o del sistema Manx, qual è la sua funzione, e, se necessario, quale opzione dev'essere specificata perché il file venga creato e/o impiegato. Si noti che in aggiunta ai file previsti dai diversi tool di programmazione, nella Tavola 2 sono riportati anche i file che si possono produrre tramite la ridirezione dell'output offerta dall'AmigaDOS, file che non erano stati mostrati nella Figura 1 e che di solito contengono i messaggi relativi agli errori che si possono verificare nell'esecuzione dei tool, oltre ovviamente ai messaggi d'inizio e fine esecuzione. Questi file possono essere esaminati con l'editor oppure stampati. La ridirezione dell'output è utile quando non si desidera impegnare la finestra di lavoro, per esempio quando si esegue un tool in background (tramite il comando RUN dell'AmigaDOS).

La ridirezione dell'output costituisce un'operazione importante nel file comandi di compilazione e link che presenteremo nel prossimo articolo: in esso i messaggi d'errore prodotti da ogni tool vengono automaticamente portati nell'editor affinché il programmatore possa velocemente esaminarli per capire se qualcosa non è andato per il verso giusto.

La Tavola 2 indica anche quali file del sistema Manx sono in caratteri ASCII e quali non lo sono. Questa informazione è importante perché ci dice

Nome del file	Posizione del file e opzioni associate	Descrizione del file
File di input dell'editor di testi Z		
ff.c	Definita dall'utente	File sorgente originale (ASCII)
z.opt	Definita dall'utente	File delle opzioni per l'editor (ASCII)
nome-qualsiasi	Definita dall'utente	File tags per le funzioni C globali di sistema
nome-qualsiasi	Definita dall'utente	File tags per le funzioni relative al programma
File di output dell'editor di testi Z		
ff.c	Definita dall'utente	File sorgente corretto o modificato (ASCII)
File di input del compilatore C		
ff.c	Definita dall'utente	File sorgente C definitivo (ASCII)
file INCLUDE	Definita dall'utente (–I)	File INCLUDE scritti in C relativi al programma (ASCII)
nome-qualsiasi	Definita dall'utente (+I)	File INCLUDE scritti in C precompilati relativi al programma
File di output del compilatore C		
nome-qualsiasi	Definita dall'utente (–O)	Nome e subdirectory per un secondo file Assembly (ASCII)
ff.sym	Definita dall'utente (+H)	Tavola dei simboli relativa al programma Assembly (ASCII)
AztecC.err	Definita dall'utente	File opzionale riportante una migliore documentazione degli errori (ASCII)
file temporaneo	Definita dall'utente	Viene creato per essere utilizzato esclusivamente dall'assembler
ff.e	Definita dall'utente (>)*	File ottenuto con la ridirezione dell'output del compilatore, utile per visualizzare gli errori (ASCII)
File di input dell'assembler		
file sorgente "asm"	Definita dall'utente	Ulteriori programmi in sorgente Assembly relativi al programma C (ASCII)
file temporaneo	Definita dall'utente	L'assembler cancella questo file dopo la lettura
file INCLUDE	Definita dall'utente (–I)	File INCLUDE scritti in linguaggio Assembly richiamati dal programma (ASCII)
File di output dell'assembler		
ff.o	La stessa del file ff.c	Codice oggetto del programma (Non-ASCII)
nome-qualsiasi	Definita dall'utente (–O)	Nome e subdirectory per un secondo modulo oggetto (Non-ASCII)
ff.lst	La stessa del file ff.c (–L)	Listato del programma in Assembly e in codice macchina (ASCII)
ff.a	Definita dall'utente (>)*	File ottenuto dalla ridirezione dell'output dell'assembler, utile per la visualizzazione degli errori (ASCII)
File di input del linker		
ff.o	La stessa del file ff.c	Codice oggetto del programma (Non-ASCII)
nome-qualsiasi	Definita dall'utente (–F)	File contenente una sequenza di opzioni per il linker (ASCII)
nome-qualsiasi	Definita dall'utente	Altri moduli oggetto collegati al programma principale (Non-ASCII)
nome-qualsiasi	Definita dall'utente (–L)	File predefiniti del sistema Manx, riguardanti le librerie linked (Non-ASCII)
File di output del linker		
ff	La stessa del file ff.c	Programma finale eseguibile (Binario)
nome-qualsiasi	Definita dall'utente (–O)	Nome e subdirectory per un secondo modulo eseguibile (Binario)
ff.sym	La stessa del file ff.c (–T)	Tavola dei simboli usati dal linker (ASCII)
ff.dbg	La stessa del file ff.c	File generato appositamente per il Source Level Debugger (Non-ASCII)
nome-qualsiasi	Definita dall'utente (–W)	File utilizzabile con il ROMWack (Non-ASCII)
ff.l	Definita dall'utente (>)*	File ottenuto dalla ridirezione dell'output del linker, utile per visionare gli errori (ASCII)
File di input del Source Level Debugger		
ff	La stessa del file ff.c	Programma eseguibile (Binario)
ff.dbg	La stessa del file ff.c	File prodotto dal linker per il Source Level Debugger (Non-ASCII)
nome-qualsiasi	Definita dall'utente (–M)	File contenente i comandi per il Source Level Debugger (ASCII)
File di output del Source Level Debugger		
nome-qualsiasi	Definita dall'utente	File contenente i comandi impartiti al Source Level Debugger e il suo output (ASCII)

* il simbolo di maggiore ">" indica che l'output del file normalmente diretto allo schermo è stato inviato a un file.

Tavola 2: sommario di tutti i file di input e di output utilizzati dal sistema di programmazione Aztec C68K della Manx

quali dei file generati dal sistema possono essere esaminati, studiati ed eventualmente stampati. Infine, mostra quali file hanno un formato studiato apposta per alcuni di questi tool di programmazione. Sotto questo profilo, si ricordi in particolare che il file comandi e i file tags relativi all'editor, il file standard AztecC.err degli errori e i file ff.asm, ff.sym, ff.lst generati dal compilatore, il file di comandi per il linker, e infine i due file di output del *Source Level Debugger*, sono tutti file ASCII.

Ciò che si deve capire è che tutti questi file, specialmente quelli ASCII, costituiscono un'ottima fonte d'informazioni per analizzare il proprio lavoro e renderlo più efficiente. L'attento esame di questi file rappresenta uno dei metodi migliori per familiarizzarsi con la programmazione in C dell'Amiga tramite il sistema Manx.

La prossima puntata

Nel prossimo articolo discuteremo l'aspetto generico di un sistema di programmazione comprendente uno o più disk drive, analizzando la migliore dislocazione dei file fra il RAM disk e i due disk drive. Verranno analizzate le estensioni del CLI come *WShell*, *ConMan*, e *ARexx*, oltre a una completa startup-sequence che ottimizza l'intero sistema, e un file comandi che rende completamene automatica la trasformazione di un file sorgente in un programma eseguibile.

Mostreremo anche come aumentare l'efficienza del proprio sistema di programmazione utilizzando opportuni file comandi che traggano vantaggio dalle caratteristiche multitasking dell'Amiga: correzione, compilazione, e fase di link di più programmi, eseguite simultaneamente.

```
main()
{
  register a;
  long     c;

  c = 1;
  for (a = 1 ; a = 10 ; a++)
    {
      c *= a;
    }
  printf ("Fattoriale di 10: %ld\n",c);
}
```

Listato 1: il modulo sorgente di un semplicissimo programma in linguaggio C che esegue il calcolo del fattoriale di 10

```
;:ts=8
;main()
;{
        public  _main
_main:
        link    a5,#.2
        movem.l .3,-(sp)
;  register a;
;  long     c;
;
;  c = 1;
        move.l  #1,-4(a5)
;  for (a = 1 ; a = 10 ; a++)
        move.l  #1,d4
.6
;    {
;      c *= a;
        move.w  d4,d1
        ext.l   d1
        move.l  -4(a5),d0
        jsr     .mulu#
        move.l  d0,-4(a5)
;    }
.4
        add.w   #1,d4
        cmp.w   #10,d4
        ble     .6
.5
;  printf ("Fattoriale di 10: %ld\n",c);
        move.l  -4(a5),-(sp)
        pea     .1+0
        jsr     _printf
        add.w   #8,sp
;}
.7
        movem.l (sp)+,.3
        unlk    a5
        rts
.2      equ     -4
.3      reg     d4
.1
        dc.b    70,97,116,116,111,114,105,97
        dc.b    108,101,32,100,105,32,49
        dc.b    48,58,32,37,108,100,10,0
        ds      0
;
;
;
        public  _printf
        public  .begin
        dseg
        end
```

Listato 2: la traduzione in Assembly sorgente svolta dal compilatore. Si noti la presenza delle originarie istruzioni in C

```
Aztec 68000 Assembler 3.6a  12-18-87
     1 0000:                    ;:ts=8
     2 0000:                              public  _main
     3 0000:                    _main:
     4 0000:  4e55 fffc                   link    a5,#.2
     5 0004:  2f04                        movem.l .3,-(sp)
     6 0006:  2b7c 0000 0001              move.l  #1,-4(a5)
       000c:  fffc
     7 000e:  7801                        move.l  #1,d4
     8 0010:                    .6

     9 0010:  3204                        move.w  d4,d1
    10 0012:  48c1                        ext.l   d1
    11 0014:  202d fffc                   move.l  -4(a5),d0
    12 0018:  4eba xxxx                   jsr     .mulu#
    13 001c:  2b40 fffc                   move.l  d0,-4(a5)
    14 0020:                    .4
    15 0020:  5244                        add.w   #1,d4
    16 0022:  b87c 000a                   cmp.w   #10,d4
    17 0026:  6fe8                        ble     .6
    18 0028:                    .5
    19 0028:  2f2d fffc                   move.l  -4(a5),-(sp)
    20 002c:  487a xxxx                   pea     .1+0
    21 0030:  4eba xxxx                   jsr     _printf
    22 0034:  504f                        add.w   #8,sp
    23 0036:                    .7
    24 0036:  281f                        movem.l (sp)+,.3
    25 0038:  4e5d                        unlk    a5
    26 003a:  4e75                        rts
    27 003c:  ffff fffc         .2        equ     -4
    28 003c:  0010              .3        reg     d4
    29 003c:                    .1
    30 003c:  4661 7474 6f72              dc.b    70,97,116,116,111,114
       0042:  6961 6c65 2064              dc.b    105,97,108,101,32,100,105,32,49
       0048:  6920 31
    31 004b:  303a 2025 6c64              dc.b    48,58,32,37,108,100,10,0
       0051:  0a00
    32 0054:                              ds      0
    33 0054:                              public  _printf
    34 0054:                              public  .begin
    35 0054:                              dseg
    36 0000:                              end
```

Listato 3: l'assembler ha trasformato il modulo sorgente Assembly nel modulo oggetto, e ha generato il file ".lst"

SPECIALISTI IN SISTEMI DESKTOP VIDEO E DESKTOP PUBLISHING

LOGITEK

Via Golgi 60 - 20133 Milano
Tel. 02/266.62.74
Fax. 02/ 23.12.90

presenta:

Novita' '89 per utenza professionale Amiga.

GENLOCK VCG-3+:
Genlock professionale per Amiga 500, 1000, 2000. Banda passante > 5,5 Mhz. Dissolvenza manuale e automatica, Inversione e sovraimpressione. Controlli separati per RGB, luminanza e contrasto. Contour control destro e sinistro. In/Out RGB e Videocomposito.
Prezzo: 2.680.000 + Iva.

VCG-3+

SCANNER
Da Tavolo - formato A4 (con hardcopy su carta) - 200 dpi - 16 toni di grigio - (L.1.390.000 + Iva)
Manuale B/N (L. 495.000 + Iva)
Manuale 16 toni di grigio (L. 817.000 + Iva)
Software OCR (trasforma qualsiasi file IFF in file ASCII (L. 190.000 + Iva)

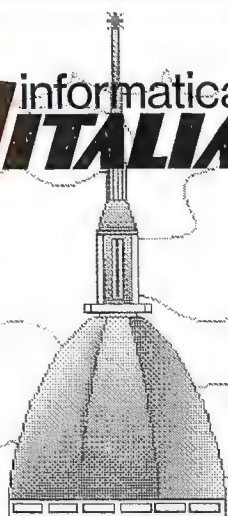
CONSULENZA - ASSISTENZA - SISTEMI COMPLETI PER TITOLAZIONE, VIDEOGRAFICA, PUBLISHING. ESPANSIONI INTERNE PER A 500/1000/2000. SERVIZIO DI STAMPA POSTSCRIPT, DIGITALIZZAZIONE, TITOLAZIONE, ANIMAZIONI, SLIDE, PROGRAMMAZIONE. SOFTWARE DI IMPORTAZIONE.

INOLTRE STRAORDINARI PREZZI PER I **SISTEMI MS-DOS**:

AT 286 TURBO - 512K RAM - 12 MHZ - HD 20 MB - HERCULES - MONITOR - DRIVE 5 1/4 1,2 MB	L. 2.280.000 + Iva
STESSA CONFIGURAZIONE - SYSTEM 30 COMPATIBILE - DRIVE 3 1/2 1,44 MB	L. 2.480.000 + Iva
STESSA CONFIGURAZIONE - 16 MHZ - 1MB RAM	L. 2.480.000 + Iva
XT TURBO - 12 MHZ - 256K RAM - MONITOR - DRIVE 5 1/4	L. 998.000 + Iva

da TORINO
in tutto il paese

IMPORTA e SVILUPPA
la Tecnologia della

COMPUTER GRAFICA
COMPUTER MUSICA

Su computer Commodore AMIGA:

*Telecamere e videoregistratori
*Digitalizzatori:
+Digiview per singoli fotogrammi
+VD Amiga in tempo reale
+FRAMER in tempo reale a colori
con encoder e memorie di quadro
*Genlock: + amatoriale PAL
+ professionale PAL Broadcast
RENDALE-VIDEOCOMP-NERIKI-MAGNI
*Software ORIGINALE per Computer Grafica
*Stampanti a colori a getto d'inchiostro
XEROX 4020
*Applicazioni musicali:
+accessori e software

INFORMATICA ITALIA S.r.l.
C.so Re Umberto, 128
10128 TORINO Tel. 011/501647

ALLA SCOPERTA DELL'AMIGA

COME PROGRAMMARE I GADGET BOOLEANI DI INTUITION

Per sfruttare al meglio i gadget booleani occorre scoprirne alcuni piccoli segreti. L'articolo analizza la gestione a mutua esclusione di tre o più gadget, l'inefficiente mutua esclusione a due gadget di SpeechToy e i gadget con maschera. Un programma dimostrativo in C

di Harriet Maybeck Tolly

Eccoci al secondo appuntamento con i gadget di Intuition. La volta scorsa abbiamo parlato dei gadget stringa, illustrandone pregi, difetti, trucchi, e concludendo con un programma in C che mostrava come impiegarli all'interno delle applicazioni. Questo articolo è dedicato ai gadget booleani ed è completato da un nuovo programma in C.

Tutti coloro che hanno selezionato le icone Retry o Cancel di un requester hanno agito su un gadget booleano. Questo tipo di gadget può fare arrivare al task due messaggi: uno quando viene selezionato (il pulsante sinistro del mouse passa da rilasciato a premuto e il puntatore è sul gadget), l'altro quando viene deselezionato (il pulsante del mouse viene rilasciato). Nella definizione del gadget si deve indicare quale messaggio dev'essere inviato (il primo, il secondo o entrambi).

Dal punto di vista dell'utente, però, è più importante il metodo scelto per mostrare sullo schermo la selezione del gadget e il suo stato di attività o di riposo. Si ricordi che i gadget booleani possono essere rappresentati con una semplice cornice, oppure (ma solo dalla versione 1.2 del sistema operativo) con un disegno più elaborato. La selezione di un gadget booleano sullo schermo può essere mostrata in molti modi diversi. Per esempio Intuition può invertirne automaticamente l'immagine e mantenerla in quello stato finché il pulsante del mouse non viene rilasciato, dopodiché il gadget torna allo stato precedente. Oppure si può definire una seconda immagine che Intuition alterna con la prima per mostrare i due stati del gadget. E così via.

Oltre al modo in cui viene segnalata la selezione, è importante anche stabilire come dev'essere mostrato all'utente lo stato del gadget. Chiameremo stato "di riposo" quello in cui il gadget non risulta selezionato, e stato "di evidenziato" l'altro. In questo senso, i gadget booleani si dividono in due grandi famiglie: i gadget hit e i gadget toggle. Vengono chiamati **gadget hit** i gadget booleani la cui immagine rimane alterata solo finché il pulsante del mouse resta premuto (nello stato di riposo mostrano sempre la stessa immagine). Sono invece **gadget toggle** quelli la cui immagine non viene alterata quando si rilascia il pulsante del mouse, ovvero quelli che Intuition cambia in modo permanente (nello stato di riposo possono mostrare l'una o l'altra delle due possibili immagini). I gadget toggle sono utili soprattutto per evidenziare situazioni di on/off che rimangono immutate fino alla nuova selezione del gadget.

I gadget booleani possono essere definiti impostando alcuni flag di attivazione nel parametro Activation della struttura Gadget. Ne esamineremo solo tre: i due che determinano il comportamento del gadget dal punto di vista del task e quello che ne regola il comportamento sullo schermo.

GADGIMMEDIATE – Se questo flag è impostato a 1, il task riceve il messaggio GADGETDOWN quando l'utente seleziona il gadget.

RELVERIFY – Se questo flag è impostato a 1, il task riceve il messaggio GADGETUP quando l'utente deseleziona il gadget, cioè quando rilascia il pulsante del mouse.

TOGGLESELECT – Se questo flag è impostato a 1, il gadget diventa di tipo toggle: quando viene selezionato, Intuition ne cambia permanentemente la rappresentazione grafica. In caso contrario (flag a 0) il gadget viene considerato di tipo hit. Si noti che questo flag, a differenza dei precedenti, non causa la generazione di messaggi.

La mutua esclusione

La mutua esclusione in un gruppo di gadget è un metodo di gestione che impone ai gadget due particolari regole di comportamento:

1) dev'essere sempre evidenziato uno e un solo gadget

2) il gadget evidenziato torna nello stato di riposo *soltanto* se l'utente ne seleziona un altro del gruppo.

Una delle maggiori critiche che vengono rivolte ai gadget booleani di Intuition riguarda proprio l'assenza della mutua esclusione, ossia l'impossibilità di definire due o più gadget in modo tale che attivandone uno si disattivi automaticamente quello che nel gruppo risultava evidenziato. La mancanza di questa possibilità è difficile da spiegare, visto che Intuition di solito fa l'impossibile per fornire meccanismi adatti a qualsiasi scopo. Comunque, come vedremo, si può aggirare l'ostacolo creando un'opportuna gestione all'interno dei propri programmi (come sempre, se Maometto non va alla montagna...).

Un esempio di mutua esclusione è costituito dai gadget per la scelta della voce maschile o femminile nel famoso demo *SpeechToy*: la voce dev'essere per forza o maschile o femminile, e non esistono altre possibilità. In *SpeechToy*, però, si ottiene la selezione del gadget in stato di riposo anche se si preme una seconda volta il pulsante del mouse sul gadget già evidenziato. Il motivo di questo inconsueto comportamento è che per la selezione del tipo di voce *SpeechToy* impiega due gadget di tipo toggle. Vediamo perché.

Molti programmatori ritengono (sbagliando) che la gestione a mutua esclusione si possa ottenere soltanto con gadget di tipo toggle. Ma la selezione di un gadget toggle già evidenziato ne provoca l'automatica conversione allo stato di riposo, e nella mutua esclusione questo implica che dev'essere automaticamente selezionato un altro gadget che si trova in stato di riposo. In altre parole, se si utilizzano i gadget toggle, nel gruppo non possono essercene più di due e questo costituisce una notevole limitazione.

A questo proposito nasce spontanea una considerazione: è vero che quando si hanno solo due possibilità che si escludono a vicenda è una scelta logica servirsi dei gadget toggle. Ma è altrettanto vero che proprio perché le possibilità sono soltanto due si potrebbe usare un solo gadget toggle, facendo in modo che il task mantenga il "ricordo" dello stato in cui si trova. Usarne due significa complicarsi la vita, tanto più che un solo toggle può mostrare sullo schermo due immagini completamente diverse, una per ciascuno stato. L'unica giustificazione per una simile scelta potrebbe essere il desiderio di fare apparire contemporaneamente sullo schermo le due alternative, come i simboli di maschio e femmina in *SpeechToy*.

A questo punto dovrebbe essere evidente che la gestione a mutua esclusione vera e propria, quella che prevede più di due gadget, non si può realizzare con i toggle. Occorre servirsi necessariamente dei gadget di tipo hit.

Descriviamo ora il metodo per ottenere la gestione a mutua esclusione con tre gadget, tenendo presente che lo stesso metodo si può applicare a qualunque numero di gadget.

La mutua esclusione con tre gadget

Perché tre gadget funzionino in mutua esclusione, occorre anzitutto che il task imposti per ognuno di essi il flag GADGIMMEDIATE, in modo che quando se ne seleziona uno il task riceva sempre il messaggio GADGETDOWN. Poi si deve decidere quale dei tre risulterà evidenziato all'inizio. A questo scopo si deve impostare per quel gadget il flag SELECTED del parametro Flags contenuto nella relativa struttura Gadget. Consultando il manuale di Intuition sembra che questo flag sia ininfluente per i gadget che non possiedono il flag TOGGLESELECT impostato, ma è un'impressione sbagliata (in particolare quando è impostato il flag GADGIMMEDIATE).

In questa fase si deve anche fare in modo che, durante la definizione dei gadget, il puntatore GadgetRender contenuto nelle rispettive strutture Gadget individui una struttura Image opportunamente inizializzata. Se invece punta a una struttura Border, oppure se non è definito (NULL), potrebbero nascere alcuni problemi a causa del metodo semplicistico con cui Intuition gestisce il refresh quando il gadget è privo d'immagine (si veda la prossima sezione).

Infine, sempre per lo stesso gadget, si deve impostare a 1 il parametro MutualExclude (attualmente questo parametro non viene mai preso in considerazione da Intuition). Questo parametro viene mantenuto aggiornato dal task stesso, e serve per documentare lo stato del gadget precedente all'ultima selezione.

Una volta che il gruppo di gadget è stato attivato, il task deve attendere da Intuition un messaggio IDCMP di tipo GADGETDOWN. In altre parole, deve attendere che l'utente selezioni un gadget. A questo punto il task deve rilevare quale gadget è stato selezionato e scoprire se si tratta di uno di quelli da gestire in mutua esclusione. Se la risposta è affermativa, occorre ancora verificare se il gadget selezionato si trova già nello stato di evidenziato, nel qual caso non si deve compiere alcuna operazione. Questo controllo può essere facilmente svolto leggendo il contenuto del parametro MutualExclude relativo al gadget. Se è impostato a 1, il task non deve eseguire alcuna operazione perché il gadget è già evidenziato. Se invece risulta impostato a 0, significa che l'utente ha selezionato un gadget in stato di riposo. In questo secondo caso, dopo aver rimosso tutti e tre i gadget il task deve azzerare i parametri MutualExclude e i flag SELECTED di tutti e tre. Questa operazione assicura che al nuovo refresh il gadget

che risultava attivato sarà visualizzato in stato di riposo. Successivamente, occorre impostare a 1 il bit di flag SELECTED e il parametro MutualExclude del gadget appena selezionato, e procedere a riattivare i gadget operandone anche il refresh.

In questo modo, si riesce facilmente a ottenere la gestione in mutua esclusione di tre gadget, e il sistema illustrato può essere banalmente esteso a un qualsiasi numero di gadget. Il programma dimostrativo che accompagna l'articolo visualizza diversi gadget, tre dei quali funzionano in mutua esclusione.

La mutua esclusione con due gadget

Per quanto riguarda invece la mutua esclusione fra due soli gadget, se volessimo riprodurre i gadget maschio e femmina di *SpeechToy* senza impiegare i gadget toggle, potremmo seguire il sistema suggerito da Jim Mackraz (autore dei cambiamenti introdotti con la versione 1.2) che permette di affidarsi esclusivamente ai gadget di tipo hit. Il vantaggio è che il programma detiene il completo controllo sui flag dei due gadget. Vediamo in che cosa consiste.

Come nel caso precedente, occorre che il task definisca entrambi i gadget con il flag GADGIMMEDIATE impostato, e che imposti il flag SELECTED per il gadget che al momento dell'attivazione dev'essere evidenziato.

Il secondo passo, come nella precedente gestione a tre gadget, consiste nel fare in modo che durante la definizione dei gadget, il puntatore GadgetRender contenuto nelle rispettive strutture Gadget individui una struttura Image opportunamente inizializzata.

Quando i gadget sono stati attivati, il task deve rimanere in attesa che giunga il messaggio GADGETDOWN. Poi, dopo aver rimosso entrambi i gadget, ci sono due possibilità: il gadget selezionato si trova nello stato di riposo o di evidenziato. Nel primo caso deve impostare il flag SELECTED di quello selezionato e cancellarlo nella struttura Gadget dell'altro. Nel secondo caso, viceversa, deve azzerare il flag SELECTED di quello selezionato e impostarlo nella struttura Gadget dell'altro. Eseguita l'una o l'altra azione, basta riattivare i gadget tramite la funzione AddGList e sottoporli al refresh tramite la funzione RefreshGList. Riassumiamo le varie fasi:

- si rimuovono i gadget il cui stato dev'essere modificato
- si impostano i flag, il che in C equivale a:
```
GadgetBooleano_uno.Flags ^= SELECTED;
GadgetBooleano_due.Flags ^= SELECTED;
```
- si riattivano i gadget
- si esegue il refresh dei gadget.

Nel modo appena descritto si riesce a ottenere lo stesso comportamento mutuamente esclusivo e auto-deselezionante previsto da *SpeechToy* senza impiegare i gadget toggle.

Il refresh dei gadget privi d'immagine

Come abbiamo detto in precedenza, quando si ricorre alla mutua esclusione con gadget di tipo hit, ognuno di essi dovrebbe possedere una struttura Image opportunamente inizializzata. Infatti, ogni variazione effettuata sul flag SELECTED, e seguita da una chiamata alla funzione RefreshGadgets, genera diversi problemi con i gadget privi della struttura Image. Se per esempio si cerca di modificare il flag SELECTED di un gadget dotato solo del bordo (struttura Border), o che non ha neanche il bordo, i colori di quel gadget mutano senza ragione apparente. È tutto va imputato al tipo di gestione che Intuition conduce per cambiare la rappresentazione dei gadget privi della struttura Image.

Con questi gadget, infatti, quando si chiama la funzione RefreshGadgets per ridisegnarli sullo schermo, Intuition va a controllare lo stato del relativo flag SELECTED. Se è impostato a 1 il gadget deve passare dallo stato di riposo a quello di evidenziato, oppure rimanere nello stato di evidenziato: insomma, dopo la chiamata dovrebbe trovarsi inequivocabilmente evidenziato. In realtà le cose non vanno proprio così, in quanto Intuition, se rileva che il flag SELECTED è impostato, effettua automaticamente il complemento di tutti i bit che rappresentano il gadget, senza preoccuparsi di verificare se era già in stato di evidenziato (nel qual caso, l'operazione di complemento a 2 dei suoi bit lo fa tornare nello stato di riposo). Allo stesso modo, se si azzera il flag SELECTED per un gadget in stato di evidenziato e si procede al suo refresh, il gadget non torna affatto nello stato di riposo. Secondo chi scrive è un bug, anche se la Commodore afferma che si tratta soltanto di una questione di documentazione.

Il problema assume proporzioni ben maggiori quando al nostro gadget associamo una struttura di tipo GadgetText. A seconda del modo grafico prescelto, certe parti del testo vengono complementate anch'esse in maniera errata. Un programma di pubblico dominio chiamato *MxGads*, su Fred Fish #52, ha cercato di risolvere il problema. Come abbiamo visto, si deve evitare che venga effettuato il complemento quando non serve, e che non venga fatto quando invece ce ne sarebbe bisogno. *MxGads* risolve il problema chiamando la funzione RefreshGadgets quando (apparentemente) non sarebbe necessario. Infatti, alla prima chiamata di questa funzione, con il flag SELECTED ancora impostato, Intuition porta il gadget nello stato di riposo. Così, quando azzeriamo il flag e chiamiamo nuovamente la funzione, il lavoro che non viene svolto questa volta lo abbiamo già fatto con la chiamata precedente. Questa sembra essere la soluzione universalmente adottata dagli addetti

ai lavori, ma non è sicuramente la migliore. Secondo noi è preferibile seguire i consigli del manuale Enhancer (l'aggiornamento dell'*Amiga Intuition Reference Manual*), e utilizzare questo flag nella mutua esclusione solo con i gadget dotati di strutture Image.

Per chi non vuole imbattersi nelle trappole del flag SELECTED, il consiglio può essere quello di usare RefreshGList. Questa routine, nata con la versione 1.2, consente di effettuare il refresh solo di alcuni gadget, in modo da non coinvolgere quelli di cui non possiamo prevedere il comportamento. Se invece volete effettuare a tutti i costi il refresh di un gadget booleano dotato di bordo e in stato di evidenziato (SELECTED a 1), almeno sappiate a che cosa andate incontro!

Gadget booleani con testo

I gadget booleani spesso includono stringhe di testo al proprio interno. Molti requester che mostrano nomi di file li usano per illustrare le possibili scelte. In questo caso assomigliano molto ai gadget stringa che abbiamo visto nel numero scorso, anche se l'utente non può scrivervi. Un punto che vorremmo mettere in evidenza è che sovrapponendo una serie di stringhe in un gadget booleano, i caratteri più a destra della stringa precedente possono ancora risultare visibili se quella che si sovrappone è più corta. Un modo per non incappare in questo guaio è aggiungere nella nuova stringa da sovrapporre tanti spazi quanti sono necessari per coprire interamente la stringa da sostituire.

Durante l'esecuzione del programma che accompagna l'articolo, appare la parola YES nel primo gadget. Quando si preme il mouse, la parola YES viene sostituita dalla parola NO. Se la S di YES non fosse stata opportunamente coperta da uno spazio ("NO "), visualizzando la stringa NO si sarebbe ottenuto NOS. Ovviamente questo è solo uno dei metodi possibili per cancellare i caratteri indesiderati; l'importante è ricordarsi di cancellarli. Nelle versioni del sistema operativo precedenti alla 1.2, quando veniva variata l'immagine di un gadget, l'eventuale testo in esso contenuto non veniva riscritto. Nella nuova versione questo bug è stato risolto.

Estensioni della versione 1.2

I gadget con maschera. Fino alla versione 1.2, un gadget di tipo booleano doveva avere il puntatore SpecialInfo azzerato. Dalla versione 1.2, invece, questo puntatore assume per i gadget booleani un significato particolare: permette di definire i cosiddetti gadget con maschera. In precedenza, i gadget booleani potevano avere soltanto la forma di rettangoli. Anche se Intuition consentiva di disegnare un'immagine all'interno del gadget, la sua area di sensibilità alla selezione del mouse era sempre delimitata dal rettangolo esterno. Questo fatto portava molto spesso all'involontaria selezione di un gadget quando il puntatore non si trovava sul suo disegno, ma appena di fianco. Ora, grazie ai gadget con maschera, è possibile specificare la forma che definisce l'area sensibile alle azioni del mouse (la maschera), in modo che corrisponda soltanto al disegno.

I passi richiesti per creare un gadget di questo tipo sono i seguenti:

- si definisce la maschera come un singolo piano di bit (bit plane)
- si definisce una struttura BoolInfo, il cui parametro Mask punta ai dati della maschera
- si imposta il parametro SpecialInfo della struttura Gadget all'indirizzo della struttura BoolInfo
- si attiva il flag BOOLEXTEND del parametro Activation: questo dice a Intuition che esiste una struttura BoolInfo puntata da SpecialInfo.

La struttura BoolInfo è così definita:

```
struct BoolInfo
{
  USHORT Flags;/* Si veda piu' avanti */
  UWORD  *Mask;/* La maschera di bit per l'evidenzia-
        mento e la selezione deve seguire le stesse
        regole impiegate per la struttura Image.
        Le sue dimensioni sono determinate da quelle
        del rettangolo di selezione del gadget */
  ULONG Reserved;/* Impostato a 0 */
};
/* Si imposti in BoolInfo.Flags
   il bit di flag che segue */
#define BOOLMASK 0x0001
/* Estensione per il gadget mascherato */
```

Senza una maschera, è possibile soltanto specificare la larghezza e l'altezza nella struttura Gadget, e quindi l'unica forma sensibile al mouse che il gadget può assumere è quella rettangolare. Tuttavia, se si inizia con un rettangolo e poi se ne maschera una parte, si possono ottenere altre forme per l'area di sensibilità del gadget.

Nell'esempio che accompagna l'articolo, è stata definita per il gadget con maschera numero 2 una struttura Image: si tratta di un ovale in colore 2, contenente all'interno un ovale più piccolo in colore 3. La maschera ha la stessa forma dell'ovale più piccolo, così da rendere sensibile solo questa parte dell'intero rettangolo occupato dal gadget.

L'immagine definita nel programma viene visualizzata come gadget. Qualsiasi selezione all'esterno dell'ovale più piccolo non viene riconosciuta da Intuition. Se invece l'utente seleziona l'ovale più piccolo, Intuition riconosce la pressione del pulsante e invia al task un messaggio di tipo GADGETUP. Inoltre, se il flag GADGHCOMP del gadget è impostato a 1, Intuition evidenzia il gadget, invertendo però solo l'ovale interno.

Sebbene le maschere siano strumenti molto utili, occorre fare attenzione a due particolari. Non è possibile accostare due maschere appartenenti a gadget diversi, in quanto dev'esserci lo spazio sufficiente per i rettangoli che le contengono. Inoltre, quando il gadget viene trasformato in fantasma (per una definizione dei gadget fantasma si veda l'articolo "Come programmare i gadget stringa di Intuition", nel numero 6/88 di *Commodore Gazette*) i Fuzzies appaiono sull'intero rettangolo che definisce il gadget, ma non sulla maschera, ovvero la parte sensibile alla selezione. Nonostante quest'inaspettata assenza, in realtà anche la maschera diventa fantasma e perde la sua sensibilità.

Vediamo ora un esempio molto semplice di mascheratura. Si deve prima di tutto impostare il parametro Mask della struttura BoolInfo all'indirizzo della matrice di dati dell'immagine, quella individuata dalla struttura Image. Così la maschera è uguale al piano di bit dell'immagine che costituisce il gadget. Questo naturalmente ha senso se l'immagine è composta da *un solo* piano di bit, ma se per l'immagine sono stati definiti più piani, non ha molto significato assegnare alla maschera un puntatore al primo piano di bit dell'immagine. Infatti i bit di ciascun piano dell'immagine rappresentano i colori, mentre i bit della maschera possono essere soltanto accesi o spenti. Se si desidera una figura colorata, servirà una maschera nella quale ogni bit è impostato a 1 se anche uno soltanto dei corrispondenti bit nei diversi piani risulta impostato (in pratica, i bit della maschera devono essere a zero solo in corrispondenza dei pixel che sullo schermo sono trasparenti).

Se invece si desidera che l'area di sensibilità sia diversa dai contorni del disegno, il problema non si presenta più, e la maschera dev'essere creata separatamente. Nel nostro programma dimostrativo, per ragioni di flessibilità abbiamo reso indipendente la maschera dall'immagine del gadget. Quindi, in realtà, non siamo obbligati a far corrispondere integralmente le strutture Image e Mask: si potrebbe per esempio definire come maschera la zona esterna all'ovale, in modo che quella diventi l'area sensibile del gadget. In effetti, è possibile definire un gadget mascherato anche se manca completamente la struttura Image.

Gadget di tipo Cancel/Retry con scelta dalla tastiera. I gadget booleani Cancel e Retry dei requester di sistema possono ora essere selezionati con le combinazioni di tasti AmigaSinistro-V e AmigaSinistro-B. Questa nuova caratteristica è resa possibile dall'aggiunta di nuovi flag alla struttura Requester. Il flag NOISYREQ indica che l'input da tastiera dev'essere filtrato quando un requester è attivo. Tale flag sui requester è impostato dalla funzione BuildSysRequest. Se si usa un requester creato con questo metodo, si possono gestire i flag VANILLAKEY e RAWKEY, e associare ai gadget booleani del requester alcune combinazioni di tasti. Così facendo, quando l'utente preme i tasti appropriati, è possibile chiamare la funzione FreeSysRequest e rimuovere il requester senza aver fatto uso dei segnali di input del mouse.

Alcuni consigli generici sui gadget

Le chiamate a OnGadget e OffGadget talvolta causano uno sfarfallio eccessivo. Ciò è dovuto al fatto che queste funzioni chiamano al loro interno la funzione RefreshGadgets. Indicando a questa funzione un particolare gadget, essa non solo effettua il refresh di quel gadget, ma anche di tutti quelli che lo seguono nella lista (cosa non sempre gradita). Questo costringe il task a collocare sempre al termine della lista i gadget per i quali è richiesto il refresh, in modo che gli altri non vengano interessati.

La versione 1.2 del sistema operativo offre però un'alternativa: la funzione RefreshGList, che permette di specificare il numero di gadget da ridisegnare. Diventa così possibile effettuare il refresh solo su un ristretto numero di gadget. Per il medesimo motivo sono nate anche AddGList e RemoveGList che permettono di specificare il numero di gadget da aggiungere o togliere dalla lista.

Si noti che AddGList consente di specificare il requester a cui appartiene il gadget. AddGadget non conteneva questo parametro, quindi d'ora in avanti sarà bene utilizzare la nuova funzione AddGList quando si aggiunge un gadget a un requester.

Il programma

Il listato che accompagna l'articolo è un semplice programma in C che apre una finestra contenente sei gadget booleani:

- un gadget di tipo hit contenente una stringa che viene sostituita da una seconda stringa alla prima selezione del gadget
- un gadget di tipo toggle con un'immagine alternativa per lo stato di evidenziato, e contenente un testo; si noti che il testo non viene interessato dal cambio d'immagine
- un gadget di tipo toggle con maschera che si può trasformare in fantasma tramite un'apposita voce del menu Gadget, o con una particolare combinazione di tasti
- una terna di gadget mutuamente esclusivi, definiti in modo che quando vengono selezionati il programma si limita a invertirli. Cioè non possiedono un'immagine alternativa per lo stato di evidenziato; si noti la particolare gestione svolta impiegando solo gadget di tipo hit.

Come di consueto, lasciamo ora la parola al programma. Nel prossimo numero concluderemo la nostra breve serie dedicata a Intuition parlando dei gadget proporzionali.

Listato: *Gadget booleani*

```
/*************************************************************
   Programma in C che mostra la gestione dei gadget booleani.

   E' stato compilato con il compilatore Aztec C della Manx,
   e necessita della versione 1.2 dell'AmigaDOS per funzionare
   correttamente; puo' esere mandato in esecuzione
   direttamente da CLI.

   N.B.: tutti i dati delle immagini devono risiedere nella
   memoria chip (i primi 512K). Se si usa il compilatore
   Lattice, prima della fase di link occorre sottoporre il
   codice oggetto all'utility ATOM; se invece si usa il Manx
   bisogna selezionare l'opzione +Cdb del linker.

   Il programma e' stato realizzato da Harriet Maybeck Tolly:
   e' di pubblico dominio e come tale puo' essere distribuito
   gratuitamente.
**************************************************************/


#include <exec/types.h>
#include <exec/exec.h>
#include <intuition/intuition.h>
#include <graphics/gfxbase.h>
#include <functions.h>

struct IntuitionBase *IntuitionBase       = NULL;
struct GfxBase       *GfxBase             = NULL;
struct Window        *FinestraDiControllo = NULL;
struct IntuiMessage  *MioIntuiMessaggio;


/**********************************************
 * Cornice del gadget hit (GadgetBooleano[0]) *
 **********************************************/


WORD   CorniceGad[]  = {0, 0, 51, 0, 51, 41, 0, 41, 0, 0};

struct Border Cornice = {
   -1,-1,        /* offset, relativi al gadget, della cornice */
   3,2,JAM1,     /* colori tratto/fondo e modo grafico */
   5,            /* numero di vettori della cornice */
   CorniceGad,   /* puntatore all'array dei vertici */
   NULL          /* non seguono altre cornici */
   };

/* I dati delle immagini devono risiedere nella memoria chip


 *******************************************************
 * Due immagini per il gadget toggle (GadgetBooleano[1]) *
 *******************************************************/


/* Forma a 'X' in colore 3, con 2 piani di bit */

UWORD PrimaImmagine[] = {
    0x0000, 0x0000, 0x0000, 0x0000,
    0x000F, 0xFF00, 0x00FF, 0xF000,
    0x0000, 0xFFF0, 0x0FFF, 0x0000,
    0x0000, 0x00FF, 0xFF00, 0x0000,
    0x0000, 0x000F, 0xF000, 0x0000,
    0x0000, 0x000F, 0xF000, 0x0000,
    0x0000, 0x00FF, 0xFF00, 0x0000,
    0x0000, 0xFFF0, 0x0FFF, 0x0000,
    0x000F, 0xFF00, 0x00FF, 0xF000,
    0x0000, 0x0000, 0x0000, 0x0000,

    0x0000, 0x0000, 0x0000, 0x0000,
    0x000F, 0xFF00, 0x00FF, 0xF000,
    0x0000, 0xFFF0, 0x0FFF, 0x0000,
    0x0000, 0x00FF, 0xFF00, 0x0000,
    0x0000, 0x000F, 0xF000, 0x0000,
    0x0000, 0x000F, 0xF000, 0x0000,
    0x0000, 0x00FF, 0xFF00, 0x0000,
    0x0000, 0xFFF0, 0x0FFF, 0x0000,
    0x000F, 0xFF00, 0x00FF, 0xF000,
    0x0000, 0x0000, 0x0000, 0x0000
    };

/* Forma a '+' in colore 2, con 2 piani di bit */

UWORD SecondaImmagine[] = {
    0x0000, 0x0000, 0x0000, 0x0000,
    0x0000, 0x0000, 0x0000, 0x0000,
    0x0000, 0x0000, 0x0000, 0x0000,
    0x0000, 0x0000, 0x0000, 0x0000,
    0x0000, 0x0000, 0x0000, 0x0000,
    0x0000, 0x0000, 0x0000, 0x0000,
    0x0000, 0x0000, 0x0000, 0x0000,
    0x0000, 0x0000, 0x0000, 0x0000,
    0x0000, 0x0000, 0x0000, 0x0000,
    0x0000, 0x0000, 0x0000, 0x0000,

    0x0000, 0x0FFF, 0xFFF0, 0x0000,
    0x0000, 0x0FFF, 0xFFF0, 0x0000,
    0x0000, 0x0FFF, 0xFFF0, 0x0000,
    0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
    0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
    0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
    0xFFFF, 0xFFFF, 0xFFFF, 0xFFFF,
    0x0000, 0x0FFF, 0xFFF0, 0x0000,
    0x0000, 0x0FFF, 0xFFF0, 0x0000,
    0x0000, 0x0FFF, 0xFFF0, 0x0000
    };

struct Image ImmagineNormale = {
    0,0,            /* coordinate vertice in alto a sinistra */
    64,10,2,        /* altezza, larghezza, # di piani di bit */
    PrimaImmagine,  /* puntatore ai dati grafici */
    0x03,           /* PlanePick  = piani di bit 1 e 2 */
    0x00,           /* PlaneOnOff = nessun piano di bit */
    NULL            /* non seguono altre immagini */
    };

struct Image ImmagineEvidenziata =
    {0,0,64,10,2,SecondaImmagine,0x03,0x00,NULL};


/*********************************************************
 * Immagine per il gadget con maschera (GadgetBooleano[2]) *
 *********************************************************/


USHORT DatiImmagineGadget[] = {
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x000F,0xF000,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
    0x0000,0x0FFF,0xFFF0,0x0000,
    0x0000,0xFFFF,0xFFFF,0x0000,
    0x0000,0xFFFF,0xFFFF,0x0000,
    0x000F,0xFFFF,0xFFFF,0xF000,
    0x000F,0xFFFF,0xFFFF,0xF000,
    0x000F,0xFFFF,0xFFFF,0xF000,
    0x00FF,0xFFFF,0xFFFF,0xFF00,
    0x00FF,0xFFFF,0xFFFF,0xFF00,
    0x00FF,0xFFFF,0xFFFF,0xFF00,
    0x0FFF,0xFFFF,0xFFFF,0xFFF0,
    0x0FFF,0xFFFF,0xFFFF,0xFFF0,
    0x00FF,0xFFFF,0xFFFF,0xFF00,
    0x00FF,0xFFFF,0xFFFF,0xFF00,
    0x00FF,0xFFFF,0xFFFF,0xFF00,
    0x000F,0xFFFF,0xFFFF,0xF000,
    0x000F,0xFFFF,0xFFFF,0xF000,
    0x000F,0xFFFF,0xFFFF,0xF000,
    0x0000,0xFFFF,0xFFFF,0x0000,
    0x0000,0xFFFF,0xFFFF,0x0000,
    0x0000,0x0FFF,0xFFF0,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
    0x0000,0x000F,0xF000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,


    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x000F,0xF000,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
    0x0000,0x0FFF,0xFFF0,0x0000,
    0x0000,0x0FFF,0xFFF0,0x0000,
    0x0000,0x0FFF,0xFFF0,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
```

```c
    0x0000,0x000F,0xF000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000
    };


USHORT Maschera[] = {
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x000F,0xF000,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
    0x0000,0x0FFF,0xFFF0,0x0000,
    0x0000,0x0FFF,0xFFF0,0x0000,
    0x0000,0x0FFF,0xFFF0,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
    0x0000,0x00FF,0xFF00,0x0000,
    0x0000,0x000F,0xF000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000,
    0x0000,0x0000,0x0000,0x0000
    };

struct Image MascheraBooleana =
    {0,0,64,30,2,DatiImmagineGadget,0x03,0x00,NULL,};

struct BoolInfo InfoGadgetBool = {
   BOOLMASK,   /* indica che il gadget e' con maschera */
   Maschera,   /* puntatore alla maschera */
   NULL
   };

/* Testi che appaiono nei gadget */

struct IntuiText Stringa_yes =
    {2,3,JAM2,5,5,NULL,(UBYTE *)"YES",NULL};

struct IntuiText Stringa_no =
    {2,3,JAM2,5,5,NULL,(UBYTE *)"NO ",NULL};

struct IntuiText Stringa_amiga =
    {2,3,JAM2,25,7,NULL,(UBYTE *)"Amiga",NULL};

/* Testi-didascalie dei gadget che appaiono nella finestra */

struct IntuiText TestoGadgetHit =
    {3,0,JAM2,10,70,NULL,(UBYTE *)"Gadget hit",NULL};

struct IntuiText TestoImmaginiAlternate =
    {3,0,JAM2,130,70,NULL,(UBYTE *)"Gadget toggle",
    &TestoGadgetHit};

struct IntuiText AltroTestoGadgetMaschera =
    {3,0,JAM2,263,76,NULL,(UBYTE *)"solo nella maschera",
    &TestoImmaginiAlternate};

struct IntuiText TestoGadgetMaschera =
    {3,0,JAM2,275,66,NULL,(UBYTE *)"Gadget sensibile",
    &AltroTestoGadgetMaschera};

struct IntuiText TestoGadgetMutEsclusivi =
    {3,0,JAM2,80,125,NULL,
    (UBYTE *)"Terna di gadget a mutua esclusione",
    &TestoGadgetMaschera};

/****************************************************
 * Strutture di tipo Gadget per tre gadget di esempio *
 ****************************************************/

struct Gadget GadgetBooleano[6] = {
   {  /* Gadget di tipo hit con cornice */

      NULL,           /* puntatore al gadget successivo */
      25,20,50,40,    /* posizione e dimensioni del rettangolo */
      GADGHCOMP,                   /* flag di controllo   */
      GADGIMMEDIATE | RELVERIFY, /* flag di attivazione */
      BOOLGADGET,     /* tipo di gadget */
      (APTR)&Cornice, /* puntatore alla cornice del gadget */
      NULL,           /* nessun'immagine alternativa */
      &Stringa_yes,   /* puntatore al testo del gadget */
      0,              /* nessuna gestione a mutua esclusione */
      NULL,           /* nessun puntatore a dati speciali */
      0,              /* nessun identificatore ID */
      NULL},          /* nessun puntatore a dati ausiliari */

   {  /* Gadget di tipo toggle, a immagine alternata */

      &GadgetBooleano[0],150,33,64,10,GADGHIMAGE | GADGIMAGE,
      GADGIMMEDIATE | RELVERIFY | TOGGLESELECT,BOOLGADGET,
      (APTR)&ImmagineNormale,(APTR)&ImmagineEvidenziata,
      &Stringa_amiga,0,NULL,0,NULL},

   {  /* Questo gadget dispone di maschera. Necessita dei dati
      BoolInfo, puntati da SpecialInfo, e che il flag BOOLEXTEND
      sia impostato tra i suoi flag di attivazione */

      &GadgetBooleano[1],308,28,64,30,GADGHCOMP | GADGIMAGE,
      GADGIMMEDIATE | RELVERIFY | TOGGLESELECT | BOOLEXTEND,
      BOOLGADGET,(APTR)&MascheraBooleana,NULL,NULL,0,
      (APTR)&InfoGadgetBool,0,NULL},

   {  /* 3 gadget mutuamente esclusivi, il primo evidenziato */

      &GadgetBooleano[2],84,105,64,10,GADGHCOMP | GADGIMAGE |
      SELECTED,GADGIMMEDIATE,BOOLGADGET,
      (APTR)&ImmagineEvidenziata,NULL,NULL,1L,NULL,0,NULL},

   {  &GadgetBooleano[3],184,105,64,10,GADGHCOMP | GADGIMAGE,
      GADGIMMEDIATE,BOOLGADGET,(APTR)&ImmagineEvidenziata,
      NULL,NULL,0,NULL,0,NULL},

   {  &GadgetBooleano[4],284,105,64,10,GADGHCOMP | GADGIMAGE,
      GADGIMMEDIATE,BOOLGADGET,(APTR)&ImmagineEvidenziata,
      NULL,NULL,0,NULL,0,NULL}

   }; /* fine dell'array GadgetBooleano[] */

struct NewWindow NuovaFinestraDiControllo = {
   20,20,       /* coordinate del vertice in alto a sinistra */
   428,145,     /* dimensioni iniziali (larghezza e altezza) */
   2,3,         /* colore dei dettagli e dello sfondo */
   GADGETUP|GADGETDOWN|CLOSEWINDOW|MENUPICK,    /* flag IDCMP */
   WINDOWDRAG|WINDOWDEPTH|WINDOWCLOSE|ACTIVATE, /* flag */
   &GadgetBooleano[5],          /* puntatore al primo gadget  */
   NULL,                        /* nessun ptr a CheckMark */
   (UBYTE *)"Gadget booleani", /* titolo della finestra  */
   NULL,        /* nessun puntatore a una struttura Screen */
   NULL,        /* nessun puntatore a una struttura BitMap */
   20, 20,      /* dimensioni minime  (non e' permesso      ) */
   320, 160,    /*   e massime        (il ridimensionamento) */
   WBENCHSCREEN /* tipo dello schermo */
   };

/*************************
 * Dichiarazione dei menu *
 *************************/

struct IntuiText testo_fantasma =
   {2,1,JAM2,5,1,NULL,(UBYTE *)"FANTASMA",NULL};

struct MenuItem voci_fantasma = {
   NULL,                   /* voce successiva */
   0,5,                    /* posizione */
   140,10,                 /* dimensioni */
   HIGHCOMP|ITEMENABLED|ITEMTEXT|COMMSEQ, /* flag */
   NULL,                   /* nessuna mutua esclusione */
   (APTR)&testo_fantasma,  /* puntatore al testo */
   NULL,                   /* evidenziazione */
   'F',                    /* comando abbreviato (Amiga-F) */
   NULL                    /* sotto-menu */
```

```
  };

struct Menu gadget_menu = {
  NULL,             /* voci successive della barra di menu */
  0,0,              /* posizione */
  60,10,            /* dimensioni */
  MENUENABLED,      /* flag */
  "Gadget",         /* puntatore al testo */
  &voci_fantasma    /* ptr alla prima voce del sotto-menu */
  };

/*********************************
 * Corpo principale del programma *
 *********************************/

main()

{
 struct MenuItem *ItemAddress();
 ULONG  Segnali, MIClass, MICode, numvoce;
 APTR   MIAddress;
 LONG   gad_pos;

 if (!(IntuitionBase = (struct IntuitionBase *)
                       OpenLibrary("intuition.library",
                       (LONG)LIBRARY_VERSION)))
   { puts("Non riesco ad aprire la libreria di Intuition");
     MiElimino();
     exit(FALSE);
   }
 if (!(GfxBase = (struct GfxBase *)
                 OpenLibrary("graphics.library",
                 (LONG)LIBRARY_VERSION)))
   { puts("Non riesco ad aprire la libreria grafica");
     MiElimino();
     exit(FALSE);
   }
 if (!(FinestraDiControllo = (struct Window *)
                 OpenWindow(&NuovaFinestraDiControllo)))
   { puts("Non riesco ad aprire la finestra di controllo");
     MiElimino();
     exit(FALSE);
   }

 /* Collega il menu alla finestra e visualizza le didascalie */

 SetMenuStrip(FinestraDiControllo,&gadget_menu);
 PrintIText  (FinestraDiControllo->RPort,
              &TestoGadgetMutEsclusivi,0L,0L);

 /* Ciclo finche' non si seleziona il gadget di chiusura */

 for (;;)  /* Attende un messaggio e lo elabora */
 {
   Segnali = Wait(1L <<
             FinestraDiControllo->UserPort->mp_SigBit);

   /* Elaborazione del messaggio inviato da Intuition */

   while (MioIntuiMessaggio=(struct IntuiMessage *)
             GetMsg(FinestraDiControllo->UserPort))

   { /* Prepara i dati per rispondere al messaggio */
     MIClass   = MioIntuiMessaggio->Class;
     MICode    = MioIntuiMessaggio->Code;
     MIAddress = MioIntuiMessaggio->IAddress;
     ReplyMsg  (MioIntuiMessaggio);

     switch (MIClass) /* Rileva il tipo di messaggio */
     {
       case MENUPICK:
         while (MICode != MENUNULL)
         {
           numvoce = ITEMNUM(MICode);
           switch  (numvoce)
           {
             case 0:
               /* Trasforma il gadget in fantasma: rimuove il
                  gadget prima di modificarlo, ne cambia lo
                  stato, lo riattiva ed esegue il refresh */

               gad_pos   = RemoveGList(FinestraDiControllo,
                             &GadgetBooleano[2],1L);
               GadgetBooleano[2].Flags ^= GADGDISABLED;
               AddGList     (FinestraDiControllo,
                             &GadgetBooleano[2],gad_pos,1L,
                             (LONG)NULL);
               RefreshGList(&GadgetBooleano[2],
                             FinestraDiControllo,
                             (LONG)NULL,1L);
               break;
           } /* Fine dell'istruzione switch */

           /* Preleva il codice successivo se sono
              state selezionate altre voci del menu */

           MICode = (ItemAddress(&gadget_menu,MICode))
                    ->NextSelect;
         } /* Fine del ciclo while */
         break;

       case GADGETUP:
         /* Controllo di rilascio sul gadget hit con testo */
         if (MIAddress == &GadgetBooleano[0])
         {
           /* Disattiva il gadget, cambia 'YES' con 'NO ',
              riattiva il gadget, e ne esegue il refresh */

           gad_pos  =  RemoveGList(FinestraDiControllo,
                         &GadgetBooleano[0],1L);
           GadgetBooleano[0].GadgetText = &Stringa_no;
           AddGList     (FinestraDiControllo,
                         &GadgetBooleano[0],gad_pos,1L,
                         (LONG)NULL);
           RefreshGList(&GadgetBooleano[0],
                         FinestraDiControllo,(LONG)NULL,1L);
         }
         break;

       case GADGETDOWN:
         /* Controlla se e' stato selezionato uno dei gadget
            gestiti in mutua esclusione */

         if (MIAddress == &GadgetBooleano[3] ||
             MIAddress == &GadgetBooleano[4] ||
             MIAddress == &GadgetBooleano[5] )
         {
           if (!(((struct Gadget *)MIAddress)->MutualExclude))
           {
             gad_pos = RemoveGList(FinestraDiControllo,
                              &GadgetBooleano[5],3L);
             GadgetBooleano[3].MutualExclude  =  0;
             GadgetBooleano[3].Flags          &= ~SELECTED;
             GadgetBooleano[4].MutualExclude  =  0;
             GadgetBooleano[4].Flags          &= ~SELECTED;
             GadgetBooleano[5].MutualExclude  =  0;
             GadgetBooleano[5].Flags          &= ~SELECTED;
             (((struct Gadget *)MIAddress)->MutualExclude)++;
             (((struct Gadget *)MIAddress)->Flags)|=SELECTED;
             AddGList     (FinestraDiControllo,
                           &GadgetBooleano[5],
                           gad_pos,3L,NULL);
             RefreshGList(&GadgetBooleano[5],
                           FinestraDiControllo,
                           (LONG)NULL,3L);
           }
         }
         break;

       case CLOSEWINDOW:
         /* Chiusura della finestra. Il programma si
            preoccupa di rispondere ai messaggi rimasti */

         while (MioIntuiMessaggio = (struct IntuiMessage *)
                   GetMsg(FinestraDiControllo->UserPort))
           ReplyMsg(MioIntuiMessaggio);
         MiElimino();
         exit(TRUE);
         break;

     }  /* switch */
   }    /* while  */
 }      /* for    */
}       /* main   */

MiElimino()
{
  /* Controlla sempre che una libreria sia
     stata aperta prima di richiuderla */

  if (FinestraDiControllo) CloseWindow (FinestraDiControllo);
  if (GfxBase)             CloseLibrary(GfxBase);
  if (IntuitionBase)       CloseLibrary(IntuitionBase);
}
```

EXECBASE: I PARAMETRI DINAMICI E LE LISTE DI SISTEMA

Oltre ai parametri statici, la struttura ExecBase, la più importante dell'intero sistema, contiene molti parametri dinamici relativi ai task in esecuzione. Analizzandoli insieme con le liste di sistema si scoprono molti segreti dell'Amiga

di Eugene P. Mortimore

Nell'articolo precedente abbiamo discusso i primi 16 parametri della struttura ExecBase (dal parametro LibNode al parametro IntVector) che per la maggior parte sono statici, ossia parametri che il sistema lascia immutati dopo l'attivazione. In questo secondo articolo dedicato alla struttura ExecBase, analizziamo i rimanenti parametri suddividendoli in tre categorie: gli ultimi parametri di gestione del sistema Exec (come IdleCount e DispCount), poi quelli strettamente legati al task che detiene il controllo (come TaskTrapCode, TaskExitCode) e infine i puntatori alle liste di sistema mantenute dall'Exec (da MemList a SoftIntList). Daremo anche uno sguardo a tre parametri (da KickMemPtr a KickCheckSum) che un programmatore esperto può utilizzare per modificare la procedura standard di attivazione della macchina con opportuni codici in RAM.

I parametri appartenenti alla prima categoria, quelli di gestione del sistema Exec, sono per lo più impiegati dalle routine interne dell'Amiga. I parametri appartenenti alla seconda categoria, quelli relativi al task in esecuzione, sono importanti perché offrono alcune informazioni sulle routine di supporto al task. Ogni volta che lo scheduler (l'algoritmo di ripartizione del tempo della CPU) manda in esecuzione un nuovo task, il sistema accede alla relativa struttura Task che lo definisce e ne copia alcuni parametri nella struttura ExecBase. Solo dopo quest'operazione gli cede il controllo. Questo aggiornamento della struttura ExecBase avviene quindi ogni volta che si verifica uno scambio di controllo fra i task.

Tramite il *Source Level Debugger* della Manx, oppure con un semplice programma in C, si può accedere a questi parametri per esaminare in dettaglio le caratteristiche del task in esecuzione.

La terza categoria di parametri raggruppa i puntatori alle strutture List che intestano le liste di sistema. Queste liste sono costituite da sequenze di nodi legati fra loro in doppia concatenazione (si possono scorrere in entrambe le direzioni). I nodi, a seconda della lista a cui appartengono, si riferiscono a message port, task, dispositivi e così via. In pratica a tutto ciò che viene aggiunto al sistema. Quando un task detiene il controllo, può scorrere queste liste ed esaminarne i nodi per ottenere informazioni sugli altri task che in quell'istante sono in attesa di ricevere il controllo, per sapere quali risorse sono disponibili nel sistema, quali aree di memoria sono disponibili per essere allocate, quali librerie e quali dispositivi sono stati aggiunti al sistema, se e in che modo le routine relative agli interrupt hardware sono state modificate, quali message port sono state allocate e molte altre informazioni. L'articolo è accompagnato da un breve programma in C che abbiamo realizzato per permettere l'analisi di queste importanti liste.

Le variabili dinamiche del sistema

I parametri della struttura ExecBase che ora prendiamo in considerazione sono chiamati "variabili dinamiche". Il nome deriva dal fatto che il software sistema le utilizza per mantenersi sempre aggiornato durante lo scambio di controllo fra i task. A causa della velocità con cui avviene lo scambio, di solito queste variabili non sono d'interesse per i programmi sviluppati dall'utente, ma sono utilizzate dal sistema per tenere sotto controllo la gestione multitasking.

Se si osservano queste variabili per mezzo del

Source Level Debugger è possibile vederle cambiare a mano a mano che i diversi task ricevono il controllo della CPU. Quasi tutte queste variabili si riferiscono al task che detiene il controllo del microprocessore.

Quale task detiene il controllo del 68000? **ThisTask** è un puntatore alla struttura Task che definisce il task in esecuzione. Di solito un task detiene il controllo per circa 64 millisecondi (1/Quantum sec), trascorsi i quali il controllo viene ceduto al task successivo. Si ricordi che è lo scheduler a stabilire quali task devono ricevere il controllo.

Le routine di sistema si servono di questo puntatore per accedere alla struttura Task relativa al task in esecuzione e rilevare quindi le sue caratteristiche. Le più importanti sono definite dai seguenti cinque parametri: TaskTrapCode, TaskExceptCode, TaskExitCode, TaskSigAlloc e TaskTrapAlloc, che esamineremo tra poco.

Per quanto tempo la CPU non ha potuto eseguire task? **IdleCount** rappresenta il numero di intervalli di tempo durante i quali la CPU non ha potuto eseguire alcun task (per esempio, quando tutti i task attivi dovrebbero impiegare lo schermo, e il sistema li blocca mentre l'utente sposta una finestra: fino a quando il pulsante sinistro del mouse non viene rilasciato, i task non ottengono il controllo, e il parametro IdleCount viene continuamente incrementato). A parte il tempo che occorre per terminare il boot (in questo periodo IdleCount continua a crescere), questo parametro viene incrementato molto di rado, in quanto è difficile che la CPU non possa mandare in esecuzione neanche un task.

Quanti scambi di controllo fra i task sono avvenuti da quando il sistema è stato avviato? **DispCount** illustra quante volte è avvenuto lo scambio di controllo fra i task dal momento in cui è stata avviata la macchina o è stato effettuato l'ultimo reset. Viene incrementato ogni volta che viene eseguita la routine privata Dispatch.

Dopo quanto tempo si verifica lo scambio di controllo fra i task? **Quantum** indica quante volte al secondo si verifica lo scambio di controllo della CPU fra i task presenti nel sistema. Normalmente vale 16.

Da quanto tempo l'attuale task è in esecuzione? **Elapsed** mantiene una misura del tempo che deve ancora trascorrere prima che il task in esecuzione perda il controllo (ricordiamo che si parla sempre del task individuato dal parametro ThisTask). Elapsed è un contatore alla rovescia che parte da 16 e termina a zero.

Quali flag di sistema risultano impostati? **SysFlags** è una variabile a 16 bit (word) contenente un insieme di flag del sistema (ogni bit rappresenta un flag). Molti di questi flag sono "privati", cioè a essi non si fa riferimento nei file INCLUDE dell'Exec. Questi particolari flag privati permettono alla Commodore di modificare il comportamento del sistema senza che venga esplicitamente reso noto al pubblico. Oltre a questi, SysFlags contiene altri quattro bit di flag che invece sono documentati nei file INCLUDE di pubblico dominio.

Il primo è SF_ALERTWACK: quando è impostato a 1 il sistema sta eseguendo le routine su ROM del debugger di sistema *ROMWack*. Questo bit di flag viene definito nel file INCLUDE alerts.h. I successivi tre bit vengono definiti nel file INCLUDE in linguaggio Assembly interrupts.i. Vediamo i loro significati.

SF_SAR (Scheduling Attention Required): se è impostato a 1 significa che il sistema deve procedere a una nuova ripartizione della CPU fra i suoi task.

SF_TQE (Time Quantum Expended): se è impostato a 1 indica che è terminato il periodo di tempo assegnato al task in esecuzione e che bisogna passare a una nuova ripartizione della CPU fra i task.

SF_SINT (Software INTerrupt): se è impostato a 1 significa che il sistema sta eseguendo una routine di interrupt software.

Si noti che questi quattro bit di flag vengono usati principalmente dalle routine interne del sistema, e che un task definito dall'utente di solito non ha interesse ad accedervi.

Per il task in esecuzione, gli interrupt sono attivati oppure no? **IDNestCnt** è la copia del parametro tc_IDNestCnt contenuto nella struttura Task relativa al task che detiene il controllo. Questo parametro è un valore interno dell'Exec e tiene conto delle condizioni di abilitazione o disabilitazione degli interrupt: viene incrementato ogni volta che un task esegue una chiamata alla funzione Disable e decrementato quando viene eseguita una chiamata alla funzione Enable. Di solito un task non ha bisogno di modificare questo parametro.

Può questo task essere sottoposto allo scambio di controllo fra i task? **TDNestCnt** è la copia del suo omologo tc_TDNestCnt presente nella struttura Task relativa al task in esecuzione, ed è anch'esso un parametro interno: la sua funzione è indicare se la gestione multitasking è attiva o no.

Questa variabile viene incrementata ogni volta che un task chiama la funzione Forbid dell'Exec, e decrementata ogni volta che viene chiamata la funzione Permit. Come per le precedenti, di solito il programmatore non ha bisogno di accedere a questa variabile.

Quale CPU e quale coprocessore matematico sono installati nel sistema? **AttnFlags** è una raccolta di 16 flag, alcuni dei quali sono considerati bit privati dalla Commodore (si giunge a questa conclusione rilevando che nei file INCLUDE non vi si fa esplicito riferimento). Ecco le definizioni dei pochi flag documentati.

AFF_68010: quando questo bit è impostato a 1, significa che nel sistema è presente un microprocessore Motorola 68010. Per ottenere l'informazione, la macchina cerca un particolare segnale

hardware e ne riflette lo stato nella suddetta variabile.

AFF_68020: quando questo bit è impostato a 1, nel sistema è presente un Motorola 68020. Come per il flag precedente, la presenza del microprocessore viene rilevata dal sistema tramite un particolare segnale hardware. Poiché i set d'istruzioni dei due microprocessori citati differiscono leggermente da quello del 68000, i programmi in grado di modificare il proprio comportamento in funzione del microprocessore possono accedere a questi flag per stabilire se utilizzare o meno le maggiori capacità del 68010 o del 68020.

AFF_68881: quando questo bit è impostato a 1, sulla scheda madre è stato installato anche un coprocessore matematico 68881. Un programma che esegue operazioni aritmetiche può controllare il valore di questo flag e decidere di utilizzare le capacità di calcolo del coprocessore, di gran lunga superiori a quelle delle routine software.

Il parametro **AttnResched** è usato internamente dalle routine dell'Exec. Ha a che fare con l'operazione di reschedule (abbandono del controllo da parte di un task fino al suo prossimo turno) del task in esecuzione. È possibile fare solo delle supposizioni sul funzionamento di questa variabile, ma sappiamo almeno che ci sono due routine private di sistema, Schedule e Reschedule, che controllano proprio l'operazione di schedule e reschedule dei task, utilizzando i valori delle loro priorità. Comunque un programmatore non dovrebbe mai trovarsi nella necessità di utilizzare questo parametro.

I prossimi quattro parametri sono di tipo APTR, un tipo di variabile che merita una breve ma precisa spiegazione. Nel file INCLUDE exec/types.h, il tipo APTR viene definito come puntatore a un altro puntatore, che a sua volta punta a una variabile di tipo char. Questa definizione apparentemente contorta è per esempio utile per individuare gli elementi di una matrice di caratteri.

Quali moduli residenti sono presenti nel sistema? **ResModules** punta all'array che elenca i moduli residenti contenuti nel sistema (nella versione 1.2.1 questo array si trova all'indirizzo 0x00C01800). Ogni elemento di questo array è un indirizzo che individua in memoria la struttura Resident associata al particolare modulo residente. In questo elenco appaiono dopo il boot i moduli che si trovano su ROM, ma un qualsiasi task può aggiungerne altri allocandoli su RAM. Ognuno di questi moduli è una collezione di routine che sono sempre pronte per essere eseguite, senza bisogno che siano caricate ogni volta da disco. È interessante notare che in questo elenco è possibile inserire moduli residenti indifferenti al reset della macchina. Per esempio, il programma *Guardian* pubblicato da *Commodore Gazette* sul numero 2/88 alloca in memoria un modulo residente (l'antivirus) che non viene disattivato con il reset della macchina (è possibile verificare che se *Guardian* è stato installato, nell'array dei moduli residenti compare l'indirizzo della relativa struttura Resident anche dopo un reset). Come vedremo, perché un modulo sia indifferente al reset, bisogna fare in modo che la sua struttura Resident compaia nella lista individuata dal parametro KickMemPtr. La struttura Resident associata a ogni modulo ne indica le caratteristiche principali, fra cui la versione, il tipo (libreria, task, dispositivo, risorsa...), la priorità, il nome e la stringa d'identificazione.

Nella versione 1.2.1, dopo il boot della macchina risultano in ROM 22 moduli residenti: il primo di questi è il cosiddetto modulo Exec, la cui struttura Resident si trova all'indirizzo 0x00FC00B6; la sua priorità è impostata a 120, il valore più alto tra le priorità assegnate a questi moduli.

Gli altri moduli sono associati alle diverse librerie e dispositivi residenti su ROM, e le loro priorità variano tra 110 e -60 (nell'array i moduli vengono elencati per ordine decrescente di priorità). L'utilità dei moduli residenti sta nel dotare il sistema di una serie predefinita di routine sempre disponibili, in modo che il programmatore non debba preoccuparsi di includerle nelle proprie applicazioni durante il link.

C'è qualche routine di trap definita per il task in esecuzione? **TaskTrapCode** è la copia del parametro tc_TrapCode relativo al task in esecuzione, e rappresenta un puntatore alla locazione di memoria dove inizia la routine di gestione delle trap previste dal 68000 (trap hardware e trap software generate dall'istruzione Assembly TRAP #N) che si possono verificare durante l'esecuzione del task.

Se il programmatore desidera una gestione personalizzata delle trap, deve assegnare un valore al parametro tc_TrapCode della struttura Task *prima* che il task venga inserito nel sistema attraverso la funzione AddTask. La lettura del parametro TaskTrapCode è consentita, ma non conviene tentare di modificarlo direttamente.

Se il programmatore non altera il suo contenuto, il parametro TaskTrapCode punta per default a un particolare indirizzo in ROM. Nella versione 1.2.1, questo indirizzo è 0x00FC2FF0: se è questo il contenuto del parametro TaskTrapCode, significa che le trap definite dal task in esecuzione vengono gestite dalle routine di sistema predefinite in ROM.

Sono state definite alcune routine per il trattamento delle exception software? **TaskExceptCode** è la copia del parametro tc_ExceptCode relativo al task in esecuzione, e rappresenta un puntatore alla locazione di memoria dove inizia la routine predisposta per il trattamento degli interrupt software. Questi interrupt software possono essere generati dalla funzione Cause dell'Exec, oppure dalle message port (una message port, quando riceve un messaggio, genera un interrupt software se è stato definito un particolare tipo di segnale dal task che l'ha creata).

Come per il parametro precedente, il programmatore deve definire questo puntatore prima che il task venga inserito nel sistema e diventi quindi attivo.

Il task in esecuzione ha definito una routine non standard per terminare la propria esecuzione? **TaskExitCode** contiene l'indirizzo di memoria in cui ha inizio una routine definita dal task in esecuzione perché venga eseguita dal sistema nel momento in cui il task perde definitivamente il controllo (nella formulazione in C del task, questa routine riceve il controllo quando viene per esempio eseguita l'istruzione exit()). Se non viene definito alcun codice, il sistema utilizzerà la routine standard di uscita, che nella versione 1.2.1 del software sistema ha inizio all'indirizzo 0x00FC1D28.

Quanti bit di segnale sono stati allocati dal task in esecuzione? **TaskSigAlloc** è la copia del parametro tc_SigAlloc della struttura Task: si tratta di una long word (32 bit), nella quale ogni bit impostato a 1 rappresenta un segnale che il task ha allocato. Un task può definire tramite la funzione AllocSignal dell'Exec fino a 16 segnali, numerati da 17 a 32: i primi 16 bit di segnale sono riservati al sistema.

Al contrario di quanto avviene per i due parametri precedenti, è il sistema a gestire la variabile tc_SigAlloc, tenendo conto (tramite le funzioni AllocSignal e FreeSignal) di quali bit di segnale vengono allocati o rimossi dal task: è quindi opportuno che questa variabile non venga modificata dal task.

Quanti flag di trap sono stati definiti dal task in esecuzione? **TaskTrapAlloc** è la copia del parametro tc_TrapAlloc della struttura Task relativa al task in esecuzione. I 16 bit di flag contenuti in questa word indicano al sistema i tipi di trap software del 68000 (TRAP #N) che sono state allocate dal task in esecuzione tramite la funzione AllocTrap, numerate da 1 a 16.

Come il parametro TaskSigAlloc, questo parametro viene automaticamente gestito dal sistema quando il task fa ricorso alle routine AllocTrap e FreeTrap.

Le liste di sistema

I prossimi otto parametri che esamineremo individuano in memoria le cosiddette liste di sistema. Queste liste vengono inizializzate e gestite dalle routine dell'Exec a mano a mano che vengono aggiunte nel sistema nuove entità (message port, librerie, dispositivi...). Ogni lista è composta da una serie di nodi disposti in doppia concatenazione. L'inserimento di nuovi nodi e la cancellazione di nodi obsoleti viene effettuata interamente attraverso le opportune routine dell'Exec. Ogni nodo è costituito da una struttura Node, all'interno della quale compaiono due puntatori ad altri nodi (il successivo e il precedente nella lista), il tipo del nodo, la sua priorità, e un puntatore a una stringa che dovrebbe indicare il nome assegnato al nodo (a questo proposito è interessante notare che le strutture che vengono allocate e rese disponibili all'intero sistema dovrebbero sempre avere un nome per essere identificabili). La priorità indicata all'interno della struttura Node svolge una funzione importante nella gestione delle liste, di qualsiasi tipo siano: stabilisce l'ordine con cui sono elencati i nodi della lista. I nodi a più alta priorità sono in testa alla lista, che prosegue in ordine di priorità decrescente. Quando l'Exec aggiunge un nodo a una lista, per sapere dove deve inserirlo va a "leggere" la priorità del nodo.

In genere le strutture Node sono sotto-strutture appartenenti ad altre strutture. Tutte le strutture che il sistema organizza in liste possiedono come primo elemento una sotto-struttura Node. Quindi, l'indirizzo di una struttura Node è anche l'indirizzo della struttura che la contiene. Per esempio, scorrendo la lista DeviceList si ottiene una serie di puntatori a strutture Node, ognuno dei quali individua in effetti la struttura Device che contiene quel nodo.

Quando un task detiene il controllo, può intraprendere alcune operazioni che modificano queste liste in maniera indiretta (per esempio, può aprire un dispositivo residente su disco). Le modifiche introdotte valgono per l'intero sistema, ed è per questo che le liste di sistema vengono anche definite globali. Il sistema continua ad aggiornarle per documentare il flusso degli eventi nel sistema.

Le funzioni dell'Exec preposte al trattamento delle liste si dividono in due categorie: quelle che aggiungono nodi a una lista (AddLibrary, AddDevice, AddIntServer, AddPort, AddResource, AddSemaphore, AddTask e FreeMem) e quelle che eliminano nodi (RemLibrary, RemDevice, RemIntServer, RemPort, RemResource, RemSemaphore, RemTask e AllocMem). FreeMem serve, in particolare, per rendere disponibili al sistema nuovi blocchi di memoria, mentre AllocMem serve per sottrarre memoria libera al sistema e allocarla per i propri scopi. Entrambe queste funzioni vanno dunque a modificare la lista MemList della memoria libera. La funzione primaria delle liste di sistema è collegare fra loro tutte le strutture della stessa classe; per esempio, elencare tutte le strutture Device nella stessa lista. In questo modo, quando si cerca una particolare struttura basta scorrere la relativa lista. È inoltre possibile, grazie a questa gestione delle liste, aggiungere nuove strutture (che l'Exec possa riconoscere) a seconda delle necessità dei task che entrano a far parte del sistema tramite la funzione AddTask.

Per esempio, un task può esaminare uno dopo l'altro gli elementi della lista di sistema TaskWait alla ricerca di un particolare processo per esaminare i parametri della relativa struttura Task. Quando identifica la struttura Node cercata, il task può considerare il puntatore individuato come puntatore alla relativa struttura Task, e procedere all'esame dei suoi parametri. Per esempio, può in questo modo ricavare informazioni sui nomi e

le priorità degli altri processi presenti nel sistema.

Nell'analisi delle liste di sistema, bisogna sempre tener presente una regola che in un sistema multitasking come l'Amiga è fondamentale: quando si accede a strutture di dati pubbliche, cioè disponibili in lettura e scrittura all'intero sistema, non è detto che un dato appena copiato sia ancora valido. Infatti, se un task effettua una copia del dato e subito dopo perde il controllo, e il nuovo processo altera il dato, quando il precedente task torna in esecuzione le sue informazioni su quel dato sono ormai obsolete. Quindi, dal momento che un task deve sempre assicurarsi di lavorare su dati certi, nell'esame delle liste di sistema bisogna prendere alcune precauzioni. Il sistema più facile è che il processo prima di scorrere una lista provveda a congelare momentaneamente la gestione multitasking affinché la lista in esame non venga alterata dagli altri task che potrebbero ricevere il controllo. Questo si ottiene tramite la funzione Forbid. Nel momento in cui il task in esecuzione ha completato la scansione della lista, deve preoccuparsi di riabilitare lo scambio di controllo fra i processi tramite la funzione Permit (si ricordi che quando si disabilita la gestione multitasking, non si devono mandare in esecuzione funzioni di I/O, come printf e read, perché l'operazione riabiliterebbe automaticamente il multitasking).

Ricordiamo infine che le liste di sistema, come tutte le altre liste di nodi, per loro natura legano insieme elementi che in memoria non sono contigui (vengono impiegate le liste e i puntatori proprio per elencare strutture di dati senza doverle spostare nella memoria). I vantaggi delle liste sono la maneggevolezza dei loro elementi e la rapidità con cui si possono inserire ed estrarre nodi.

Quali aree di memoria sono disponibili nel sistema? **MemList** è una lista di sistema che riporta informazioni sui blocchi (chunk) di memoria liberi nel sistema. Il fatto che la memoria libera sia suddivisa in aree non contigue non deve stupire; infatti, in un sistema multitasking le richieste di memoria da allocare e le restituzioni di memoria al sistema provengono da tutta la comunità dei task in esecuzione. Questo significa che il sistema, a mano a mano che nuovi task allocano e liberano memoria, si trova ad avere blocchi liberi di memoria sempre più spaiati. Ecco quindi che nasce la necessità di mantenere continuamente aggiornato un elenco dei blocchi di memoria liberi, in modo che in qualunque istante il sistema possa rilevare lo stato della memoria per soddisfare le richieste dei processi.

La struttura MemList elenca una serie di strutture MemHeader. Ognuna di queste strutture controlla una precisa area della memoria (per default, su un Amiga dotato di 1 MB di RAM la prima struttura MemHeader controlla la memoria fast, mentre la seconda controlla la memoria chip), della quale indica il nome, l'indirizzo superiore, inferiore, il tipo, e la quantità di byte liberi. Oltre a questi parametri, la struttura MemHeader contiene un puntatore a una struttura MemChunk. Questa struttura intesta una lista semplice (cioè priva di doppia concatenazione e quindi scandibile solo in un senso) che elenca una serie di altre strutture MemChunk. Ogni struttura di questa lista contiene un puntatore alla successiva struttura MemChunk e una variabile di tipo long che indica la quantità di byte presenti in quel blocco. Nel file INCLUDE memory.h è contenuta la definizione di questa struttura.

La lista MemList viene solitamente consultata dalle funzioni relative alla gestione della memoria, come AllocMem, FreeMem, Deallocate, Allocate, AllocEntry e FreeEntry. I blocchi di memoria elencati dalla lista formano il cosiddetto pool della memoria libera: ciascun blocco può trovarsi in qualsiasi punto della RAM disponibile nell'Amiga. Si ricordi che i primi 512K sono definiti RAM chip, mentre i byte superiori fanno parte della RAM fast (la memoria fast a differenza della memoria chip non è accessibile ai chip custom dell'Amiga).

La memoria viene sottratta al sistema per mezzo delle funzioni AllocMem, AllocEntry e Allocate, mentre viene rilasciata per mezzo delle funzioni FreeMem, Deallocate e FreeEntry. Quando un task ha bisogno di un certo quantitativo di memoria e tenta di ottenerla attraverso le routine dell'Exec, queste ricercano lo spazio richiesto tra i blocchi descritti in queste liste semplici, e quindi lo allocano.

I task hanno la possibilità di riservare una parte della memoria libera per i propri scopi, assumendosene la gestione. Per far questo, un processo deve chiamare la funzione AddMemList, la quale aggiunge alla lista MemList una nuova struttura MemHeader. Quest'ultima mantiene poi la lista dei blocchi liberi relativi alla nuova partizione della memoria, di esclusivo dominio del task che l'ha richiesta. Quando il processo ha chiamato AddMemList, sarà poi suo compito mantenere integra la lista semplice dei blocchi di memoria liberi. È in questo caso che il task utilizzerà le funzioni Allocate e Deallocate per gestire i singoli quantitativi di RAM.

Quali risorse hardware sono attualmente presenti nel sistema? **ResourceList** è una lista di strutture la cui composizione varia a seconda del tipo di risorsa. Comunque, tutte le diverse strutture contengono come primo elemento una struttura Library. Quindi i primi parametri di ogni struttura sono quelli previsti dalla struttura standard Library, mentre i successivi dipendono dal tipo di risorsa. Le risorse riconosciute dall'Amiga sono la CIA, la Potgo, la Disk e la Miscellaneous. La definizione della struttura Resource è contenuta nel file resource.h. Gli elementi della lista ResourceList vengono aggiunti e rimossi tramite le funzioni AddResource e RemResource.

Quali dispositivi sono residenti in memoria RAM e ROM? **DeviceList** elenca le strutture Device pre-

senti in RAM: si tenga presente che la struttura Device è identica alla struttura Library, e definisce sempre un dispositivo che si trova in memoria; la sua definizione si trova nel file INCLUDE exec/devices.h.

Ogni struttura Device presente nella lista indica che il relativo dispositivo si trova in memoria ed è utilizzabile. Questa informazione ha importanza soprattutto per i dispositivi non residenti. Infatti, se uno di questi dispositivi non è presente in memoria, nella lista DeviceList la sua struttura Device non compare. Quindi le strutture Device presenti in questa lista corrispondono a tutti i dispositivi che nel particolare istante si trovano in memoria (questo non significa comunque che questi dispositivi siano stati aperti da un task). Se un task chiama la funzione OpenDevice per aprire un dispositivo, e il sistema rileva che nella lista DeviceList la relativa struttura Device non è presente, ne deduce che il dispositivo è di tipo non residente e in quel momento non si trova in memoria. Procede allora a caricarlo da disco e ad aggiungere nella lista la relativa struttura Device tramite la funzione AddDevice. Da quel momento in poi, il dispositivo rimane in memoria fino a quando non viene disallocato. Anche se il task che l'ha aperto procede a chiuderlo con la funzione CloseDevice, in realtà il dispositivo continua a essere presente in memoria. L'unico modo per cancellarlo dalla memoria è disallocarlo con la funzione RemDevice, dopo averlo chiuso.

Attualmente l'Amiga riconosce 12 dispositivi (dal dispositivo Audio al dispositivo TrackDisk). Il programmatore può comunque realizzare dispositivi personalizzati e inserirli nel sistema aggiungendo le relative struttura Device alla lista DeviceList tramite la funzione AddDevice.

Quali vettori di interrupt hardware sono stati modificati? **IntrList** è una lista di strutture Interrupt. In generale, la struttura Interrupt individua i dati e i codici relativi alla routine di gestione di un particolare interrupt hardware o software (quest'ultimo viene spesso chiamato exception), ed è definita nel file INCLUDE exec/interrupts.h. La lista IntrList elenca solo le strutture Interrupt relative agli interrupt hardware che non vengono gestiti in maniera standard: in particolare, gli interrupt hardware che vengono gestiti dagli interrupt server.

Ogni chiamata alle funzioni AddIntServer e RemIntServer altera la quantità di nodi nella lista, analogamente a quanto accade per le liste già descritte.

Quali sono le librerie presenti in RAM? **LibList** contiene l'elenco delle strutture Library presenti in RAM. Ognuna di queste individua una particolare libreria shared dell'Amiga: al momento, l'Amiga riconosce sette librerie predefinite (dalla libreria Exec alla libreria Intuition). Alcune sono residenti in ROM, mentre altre si trovano su disco. L'utente può creare le proprie librerie personali usando la funzione MakeLibrary, e renderle note al sistema tramite la funzione AddLibrary. Il file exec/library.h contiene la definizione della struttura Library. AddLibrary e RemLibrary sono le routine di gestione della lista LibList.

Quali message port sono presenti nel sistema? **PortList** è una lista di strutture MsgPort, ciascuna delle quali definisce una particolare message port (si ricordi che una message port appartiene sempre a uno e un solo task): queste message port consentono ai task di comunicare con gli altri task, con il sistema o con i dispositivi, scambiando messaggi e segnali che possono, tra l'altro, provocare anche l'attivazione di interrupt software. Il meccanismo di scambio dei segnali è un modo per ottenere comunicazioni asincrone tra i task. La lista PortList viene gestita attraverso le funzioni AddPort e RemPort, che permettono di aggiungere e sottrarre message port al sistema. Si noti che nella lista PortList appaiono normalmente solo le message port pubbliche, cioè quelle che sono a disposizione di chiunque desideri comunicare con i relativi task. Se un task alloca una message port soltanto per comunicare con un dispositivo, non la inserisce nella lista PortList. Se invece un task aggiunge una propria message port alla lista di sistema, questa diventa una porta pubblica: di conseguenza, qualsiasi altro task può mandare messaggi a quella porta. Questo meccanismo è alla base del sistema di comunicazione fra i processi previsto dal pacchetto *ARexx*, tramite il quale i programmi che se ne servono possono scambiarsi informazioni seguendo un particolare standard.

Quali task sono pronti per essere eseguiti? **TaskReady** è un elenco di strutture Task, ciascuna delle quali definisce un task che si trova in attesa di ottenere il controllo della CPU. Si tratta dunque di un elenco di tutti i processi che non sono "addormentati" in attesa di qualche evento (per esempio, dell'arrivo di un messaggio). In questa lista non appaiono quindi le strutture Task relative a quei task che pur essendo eseguibili, non ricevono per il momento il loro turno di controllo della CPU. Generalmente, un task viene identificato come eseguibile subito dopo la chiamata ad AddTask. È importante ricordare che un processo può trovarsi in uno dei seguenti stati: aggiunto al sistema (added), pronto per l'esecuzione (ready), in esecuzione (running), in attesa di eventi esterni (waiting), in fase di esecuzione di un interrupt software o infine rimosso dal sistema.

Oltre alla lista dei task pronti per l'esecuzione, esiste anche la lista dei task in attesa, mentre per gli altri possibili stati che abbiamo elencato non esistono liste di sistema dedicate. Del resto, il sistema non ha bisogno, per esempio, di una lista dei task che detengono il controllo della CPU, visto che con un solo microprocessore esiste sempre uno e un solo processo in esecuzione.

Le routine dell'Exec utilizzano la lista TaskReady per fornire allo scheduler della CPU le informazioni relative alle priorità dei task in attesa di ricevere cicli-macchina. Questa priorità è indi-

cata dal parametro ln_Pri della sotto-struttura Node presente nella struttura Task relativa a ogni task. Analizzando questa lista, lo scheduler della CPU può decidere di volta in volta qual è il task che deve ricevere il controllo della CPU. Nella maggioranza dei casi, il sistema operativo legge semplicemente la priorità di ogni task eseguibile, e manda in esecuzione quello con la priorità più alta. Si ricordi che se questo task non entrasse mai in stato di attesa, nessun altro task potrebbe ricevere il controllo. Fortunatamente, proprio i task a più alta priorità sono quelli che più spesso entrano in stato di attesa, e quando questo accade, lo scheduler cede il controllo al task di priorità immediatamente inferiore. Se diversi processi hanno la stessa priorità, lo scheduler li manda in esecuzione uno dopo l'altro con la temporizzazione imposta dal parametro Quantum.

I nodi della lista TaskReady vengono inseriti e rimossi tramite le funzioni AddTask e RemTask. Inoltre le routine dell'Exec aggiungono automaticamente un nodo a questa lista quando un task è in attesa di un particolare evento e l'evento si verifica. Il task passa dallo stato di attesa allo stato di eseguibile, e viene inserito nella lista TaskReady con la sua priorità.

Quali task sono in stato di attesa? **TaskWait** è ancora una lista composta da strutture Task, con la differenza che elenca i task in attesa di qualche evento esterno, come un segnale proveniente da un altro processo. Sono task che originariamente sono stati caricati da disco e aggiunti al sistema, hanno ricevuto il controllo, e dopo alcune istruzioni sono entrati in stato di attesa quando la CPU ha eseguito al loro interno una particolare istruzione; per esempio la chiamata alla funzione Wait per attendere uno o più segnali. Quando si verifica l'evento atteso dal task, il sistema automaticamente estrae la sua struttura Task dalla lista TaskWait, e la inserisce nella lista TaskReady, di modo che lo scheduler della CPU possa mandare in esecuzione il task alla prima occasione.

Quindi, la struttura TaskWait elenca i task che stanno attendendo di tornare eseguibili in seguito a una ben precisa richiesta che hanno inoltrato (un segnale, un output, un input...). Si ricordi che la priorità dei task che entrano in attesa e tornano eseguibili rimane immutata.

Quali semafori sono stati aggiunti nel sistema? **SemaphoreList** è una lista delle strutture Semaphore presenti nella memoria dell'Amiga. Una struttura Semaphore contiene tutte le informazioni relative a un particolare semaforo: i semafori vengono utilizzati per migliorare lo scambio di controllo fra i task, nell'ambito di un gruppo di task che accedono alle stesse risorse. Per la definizione della struttura Semaphore si può consultare il file exec/semaphores.h.

La funzione dei semafori è analoga a quella delle funzioni Forbid e Permit: evitare cioè che due o più task abbiano accesso agli stessi dati nello stesso periodo, e che uno dei task alteri dati che l'altro ritiene acquisiti. L'unica differenza è che i semafori consentono di circoscrivere in maniera più selettiva le modalità di disabilitazione della gestione multitasking, ossia permettono di sospendere dallo scambio di controllo multitasking soltanto alcuni task e non tutti (come avverrebbe con le funzioni Forbid e Permit).

Si noti che la lista SemaphoreList non appare di seguito alle altre, nella struttura ExecBase, ma si trova dopo i parametri VBlankFrequency e PowerSupplyFrequency. Questa posizione è senz'altro dovuta al fatto che la gestione dei semafori è stata aggiunta al sistema solo in un secondo tempo, cioè quando i progettisti decisero che le prestazioni delle funzioni Permit e Forbid erano troppo limitanti.

I nodi della lista SemaphoreList vengono gestiti attraverso le funzioni AddSemaphore e RemSemaphore.

Quali interrupt software sono stati aggiunti al sistema? **SoftInts[5]** è un array di cinque sotto-strutture SoftIntList contenute una dopo l'altra nella struttura ExecBase. Ognuna di esse contiene una struttura List e un parametro di 2 byte per mantenere l'allineamento con le long word. Per ogni struttura SoftIntList, il parametro sh_List individua una lista doppia di strutture Interrupt, che definiscono le routine di interrupt software aggiunte al sistema dai task. Esistono quindi cinque liste di strutture Interrupt, una per ognuno dei cinque livelli di priorità ammessi dall'Exec per gli interrupt software (−32, −16, 0, +16, +32). Queste strutture di tipo SoftIntList sono di pertinenza esclusiva dell'Exec e vengono usate per la gestione interna degli interrupt prodotti dalla funzione Cause o dall'arrivo di un messaggio a una message port definita per generare segnali.

Ciascun task può definire, nel momento in cui viene attivato, una o più routine di interrupt software specificando i parametri is_Code e is_Data di una struttura Interrupt. Oltre a questi due dati, il task deve anche indicare la priorità che desidera assegnare alla routine di interrupt; per far questo deve scegliere uno dei cinque valori ammessi dall'Exec e memorizzarlo nel parametro ln_Pri della sotto-struttura Node contenuta nella struttura Interrupt. Le liste che elencano in ordine di priorità le strutture Interrupt aggiunte dai task, vengono costantemente aggiornate dall'Exec. Così, quando il sistema di gestione multitasking può mandarne una in esecuzione, va a controllare i livelli di priorità per decidere quale sarà effettivamente eseguita. In generale un task definito dal programmatore non avrà mai bisogno di considerare questa variabile, che serve soltanto al sistema. Per l'esame dettagliato della struttura SoftIntList, si veda il file INCLUDE exec/interrupts.h.

Qual è il codice del messaggio relativo all'ultimo alert di Guru Meditation? **LastAlert** è un parametro, lungo 16 byte (quattro long word), che riporta le caratteristiche dell'ultimo alert di Guru Meditation verificatosi nel sistema. I tipi di alert si

dividono in due categorie: fatali ("dead-end") e non-fatali ("nondead-end"). Per esempio, un errore nel calcolo del checksum di qualche libreria (indicatore del fatto che una libreria di sistema ha subito un'alterazione), un errore nel calcolo del parametro ChkSum, un valore errato in LowMemChkSum (che segnala un'alterazione nei vettori di gestione delle exception del 68000) sono tutte condizioni che portano il sistema a un alert fatale, cioè non recuperabile. Invece, se un task cerca di aprire una libreria, per esempio la libreria di Intuition, senza riuscirvi, si verifica un alert non-fatale. In genere un alert fatale costringe a un reset della macchina, mentre un alert non fatale permette di recuparare la situazione.

Ogni messaggio relativo a un alert di Guru Meditation è composto di quattro sottocampi: una campo di una cifra che indica se si tratta di errore fatale o meno; un campo di due cifre che riporta il codice d'identificazione dell'alert; un campo di quattro cifre che indica una prima classificazione generica del tipo d'errore che ha causato l'alert; e infine un campo di quattro cifre che indica con maggior precisione il tipo d'errore. Si noti che la terza e la quarta long word dell'array LastAlert contengono i valori che appaiono sullo schermo nel tipico messaggio di Guru Meditation. Si veda il file exec/alerts.h per un elenco dei codici che documentano gli alert.

Se si verifica un alert di Guru Meditation durante l'esecuzione di un task, il debugger (o lo stesso task) può leggere il parametro LastAlert per scoprire le ragioni dell'alert. Subito dopo, a seconda del tipo di alert, il processo può eventualmente riprendere il controllo della situazione, e garantire l'incolumità dei dati che sta trattando.

Qual è la frequenza di sincronismo verticale dello schermo? **VBlankFrequency** indica la frequenza di sincronismo verticale in uso, ossia la frequenza alla quale si verificano gli interrupt relativi al blank di schermo (il blank di schermo è il periodo di tempo durante il quale il pennello elettronico viene spento e riportato nella posizione iniziale per disegnare un nuovo quadro). Ce ne sono 50 al secondo nel sistema PAL europeo e 60 al secondo nel sistema NTSC americano. Un task opportunamente predisposto può controllare questo valore per stabilire su che tipo di macchina è in esecuzione e regolare di conseguenza le sue temporizzazioni interne.

Qual è la frequenza di rete? **PowerSupplyFrequency** è un parametro che, come il precedente, tiene conto delle possibili differenze hardware nel computer in uso. Viene impiegato, tra l'altro, per regolare con una certa accuratezza alcune temporizzazioni.

Sono presenti in RAM moduli da eseguire durante il boot della macchina? **KickMemPtr** è un parametro che punta a una lista di strutture MemList, ognuna delle quali identifica un elenco di strutture MemEntry. A sua volta, ogni struttura MemEntry individua l'indirizzo di un blocco di memoria e la sua lunghezza. In genere questo indirizzo individua la struttura Resident relativa a un modulo residente su RAM che deve ricevere il controllo durante il boot della macchina. Si tratta di moduli che vengono allocati nella memoria di sistema a particolari indirizzi attraverso la funzione AllocAbs (questa funzione, tra l'altro non documentata, serve per allocare memoria a un particolare indirizzo assoluto). Una volta allocati, questi moduli di memoria vengono protetti, ossia non sono alterabili neanche con il reset della macchina. Per esempio, scorrendo l'elenco delle strutture MemEntry, potremmo trovare come ultimo modulo un antivirus del tipo di *Guardian*... oppure un maledetto virus.

Di norma il boot dell'Amiga viene eseguito attraverso codici memorizzati in ROM. In certe condizioni il programmatore può voler scavalcare i codici standard per personalizzare il comportamento della macchina. Per esempio, negli Amiga 500 e 2000, i dati necessari al boot del sistema si trovano nei primi 256K di ROM. Per l'Amiga 1000, invece, questi codici, ossia il *Kickstart*, vengono caricati da disco, rendendo il boot più flessibile. Comunque, il parametro KickMemPtr consente una flessibilità ancora superiore a quella data dal *Kickstart* su disco, perché permette di usare codici scritti in memoria RAM. In questo modo si superano le limitazioni del *Kickstart* su disco (che non è molto facile da modificare) e anche in ROM (che per essere alterato richiede la sostituzione dei chip di memoria).

Se sono presenti, dove si trovano i codici per il boot da RAM? **KickTagPtr** punta a un array di puntatori a strutture Resident. Se tutte le chiamate alla funzione AllocAbs eseguite per allocare le funzioni di boot in RAM hanno avuto successo, questo parametro viene impostato all'indirizzo della prima struttura Resident. Questo array serve a raccogliere tutti i dati relativi alle funzioni di attivazione del sistema non standard.

I codici per il boot da RAM sono integri? **KickCheckSum** è un parametro che serve per verificare che i moduli residenti inseriti per eseguire un boot non standard siano quelli previsti. Infatti, dopo un reset bisogna sempre assicurarsi che la procedura di boot in RAM sia intatta, prima di cederle il controllo.

È possibile espandere la struttura ExecBase per usi futuri? **ExecBaseReserved[10]** è un gruppo di dieci byte attualmente inutilizzati. Si tratta di uno spazio che in futuro potrà essere impiegato per aggiornare ed espandere il sistema Exec. La struttura ExecBase viene automaticamente allocata in memoria dalle routine di sistema indicando l'opzione MEMF_CLEAR, e quindi questi dieci byte contengono soltanto zeri.

ExecBaseReserved[20] è un secondo insieme di parametri, questa volta di 20 byte, previsti per impieghi futuri. Analogamente ai dieci precedenti, il sistema inizializza questi 20 byte con degli zeri.

Nel prossimo numero

Nel prossimo articolo di questa serie illustreremo i programmi *ARexx* e *WShell*. L'*ARexx* è la versione per l'Amiga del sofisticato interprete di file batch sviluppato dalla IBM, il *Rexx*.

Passando all'Amiga, questo linguaggio ha subito diverse modifiche, essenzialmente per sfruttare la capacità multitasking e la comunicazione a messaggi offerte dal sistema operativo di questo computer. I programmi dotati di message port di tipo *ARexx* possono comunicare fra loro e controllarsi a vicenda, cioè ricevere comandi da un altro task anziché dall'utente. Si tratta di possibilità già sfruttate da alcuni programmi apparsi recentemente. Diventa quindi possibile, per esempio, avere un editor di testi il cui funzionamento è controllato da un file batch *ARexx*: *CygnusEd Professional* e *TxEd Plus*, per esempio, sono programmi in grado di ricevere comandi da file batch *ARexx*.

Esamineremo poi il programma *WShell*, una particolare shell. Si tratta di un'interfaccia linea comandi migliore della macchinosa CLI: il programma *WShell*, per mezzo delle capacità di piping (ridirezione dell'output di un programma a un altro), dei comandi residenti, della gestione degli alias (sinonimi), supera molte limitazioni del CLI, calando l'utente in un ambiente di lavoro più flessibile, che può essere facilmente personalizzato e adeguato alle proprie esigenze.

Listato: *Liste di sistema*

```
/**************************************************************
  Questo programma permette di scandire alcune liste e array
  di sistema accedendo ai parametri della struttura ExecBase.
  Con il compilatore Aztec C della Manx l'oggetto
  dev'essere sottoposto alla fase di link con il comando:
     ln NomeFile -lc -ls
  Il programma per brevita' non disabilita la gestione
  multitasking tramite le funzioni Forbid() e Permit()
 **************************************************************/

#include <exec/execbase.h>
#include <exec/memory.h>
#include <exec/libraries.h>
#include <exec/tasks.h>
#include <exec/resident.h>

struct ExecBase  *SysBase;
struct Node      *NodePtr;
struct MemHeader *MemP;
struct Library   *LibPtr;
struct Task      *TaskPtr;
char        more, *Tipo [17] = {
 "UNKNOWN","TASK","INTERRUPT","DEVICE","MSGPORT","MESSAGE",
 "FREEMSG","REPLYMSG","RESOURCE","LIBRARY","MEMORY","SOFTINT",
 "FONT","PROCESS","SEMAPHORE","SIGNALSEM","BOOTNODE"};

void  MostraBlocchi(p) /* Esamina la lista semplice */
struct MemChunk *p;    /* di strutture MemChunk     */
{
   int i = 0;
   while (p != NULL) {
     printf(" Blocco%4d: %91d byte liberi\n",++i,p->mc_Bytes);
     p = p->mc_Next;
   }
   printf("Premi un tasto...\n");
}

void MostraModuliRes() { /* Esamina l'array di strutture */
   APTR   Ptr;           /* Resident relative ai moduli  */
   struct Resident *Res; /* residenti presenti           */
   Ptr = SysBase->ResModules;
   scr_clear();
   while (*Ptr != NULL) {
      Res = (struct Resident *) *Ptr;
      printf("Nome del modulo  %s\n",Res->rt_Name);
      printf("Identificazione  %s\n",Res->rt_IdString);
      printf("Versione         %d\n",Res->rt_Version);
      printf("Tipo             %s\n",Tipo[Res->rt_Type]);
      printf("Priorita'        %d\n\n",Res->rt_Pri);
      read(more); Ptr++;
   }
}

main() {
   char a[10];
   int  loop;
   for (;;) {
      do {
         loop = 0; scr_clear();
         printf("\n\n\n\n\nListe di sistema\n\n");
         puts("    1) Memoria libera    (MemList)");
         puts("    2) Risorse           (ResourceList)");
         puts("    3) Dispositivi       (DeviceList)");
         puts("    4) Interrupt         (IntrList)");
         puts("    5) Librerie          (LibList)");
         puts("    6) Message port      (PortList)");
         puts("    7) Task ready        (TaskReady)");
         puts("    8) Task wait         (TaskWait)");
         puts("    9) Moduli residenti  (ResModules)");
         puts("       ^Questa non e' una lista^");
         printf("\nQuale vuoi scandire? ");
         gets(a); scr_clear();
         switch(atoi(a)) {
            case 1: NodePtr = SysBase->MemList.lh_Head;
                    break;
            case 2: NodePtr = SysBase->ResourceList.lh_Head;
                    break;
            case 3: NodePtr = SysBase->DeviceList.lh_Head;
                    break;
            case 4: NodePtr = SysBase->IntrList.lh_Head;
                    break;
            case 5: NodePtr = SysBase->LibList.lh_Head;
                    break;
            case 6: NodePtr = SysBase->PortList.lh_Head;
                    break;
            case 7: NodePtr = SysBase->TaskReady.lh_Head;
                    break;
            case 8: NodePtr = SysBase->TaskWait.lh_Head;
                    break;
            case 9: MostraModuliRes();
                    loop++; break;
            default: scr_clear(); exit(0);
         }
      } while (loop);

      while (NodePtr->ln_Succ != 0) {
       printf("Nome del nodo  %s\n",NodePtr->ln_Name);
       printf("Tipo           %s\n",Tipo[NodePtr->ln_Type]);
       printf("Priorita'      %d\n",NodePtr->ln_Pri);
       switch(atoi(a)) {
        case 1: MemP = (struct MemHeader *) NodePtr;
         printf("Attributo        %d\n",MemP->mh_Attributes);
         printf("Inizio           0x%lx\n",(long)MemP->mh_Lower);
         printf("Fine             0x%lx\n",(long)MemP->mh_Upper);
         printf("Byte liberi      %ld\n",MemP->mh_Free);
         printf("Elenco dei blocchi (chunk):\n");
         MostraBlocchi(MemP->mh_First); break;
        case 3:
        case 5: LibPtr = (struct Library *) NodePtr;
         printf("Versione         %d\n",LibPtr->lib_Version);
         printf("Revisione        %d\n",LibPtr->lib_Revision);
         printf("Identificatore %s\n",LibPtr->lib_IdString);
         printf("In uso da %d task\n\n",LibPtr->lib_OpenCnt);
         break;
        case 7:
        case 8: TaskPtr = (struct Task *) NodePtr;
         printf("Stato del task %d\n\n",TaskPtr->tc_State);
         break;
       }
      NodePtr = NodePtr->ln_Succ; read(more);


    }
    printf ("Fine della lista. Premi un tasto...\n");
    read(more);
   }
}
```

LE MACRO ISTRUZIONI ASSEMBLY DELL'AMIGA

Le macro istruzioni sono uno strumento di programmazione in grado di far risparmiare tempo e fatica. Per l'Amiga sono già pronte molte macro Assembly. Bisogna però sapere cosa fanno e come usarle

di Eugene P. Mortimore

Analizzando un listato in linguaggio Assembly, una delle cose che saltano subito all'occhio è l'uso massiccio di macro istruzioni. Sebbene le macro sveltiscano in maniera significativa il lavoro del programmatore e contribuiscano a ridurre le dimensioni dei file sorgente, d'altra parte possono lasciare molto perplesso chi vuole studiare il programma. Infatti molte istruzioni appaiono del tutto oscure se non si conoscono le espansioni di tutte le macro impiegate.

Lo scopo di questo articolo è chiarire cosa sono le macro istruzioni e a che cosa servono. Verrà dato particolare risalto alle macro che riguardano rispettivamente i nomi e la gestione delle librerie shared. Si tratta di macro definite qua e là nei file INCLUDE e che ricorrono specialmente nei sorgenti scritti dalla Commodore. Dopo aver analizzato le espansioni di questi due gruppi di macro, diventerà molto più facile leggere i listati Assembly scritti da altri programmatori e redigere i propri in maniera più spedita.

Le macro istruzioni

In generale, nei linguaggi compilati, cioè i linguaggi che prevedono la trasformazione di un sorgente in linguaggio macchina, una macro istruzione è la definizione di un blocco di codici con un nome. In seguito, ogni volta che nel sorgente compare il nome assegnato, questo viene sostituito dal relativo blocco di codici. È il cosiddetto preprocessore (particolare sotto-programma di ogni compilatore) che, oltre ad assolvere molti altri compiti, si preoccupa durante la lettura del sorgente di espandere con le relative definizioni tutti i nomi di macro che incontra. Si comprende subito, quindi, che una macro non è una routine del programma che viene chiamata in run-time quando occorre, ma un blocco di codici sorgente che viene duplicato ogni volta che nel sorgente compare un certo nome. Le macro sono quindi, secondo questa definizione, oggetti molto generici e molto semplici a cui si può fare ricorso con qualsiasi linguaggio compilato, come il C, l'Assembly, il Pascal, il Basic non interpretato.

Nella realtà le cose si complicano un po'. Tutti i preprocessori che si rispettano consentono di passare alcuni argomenti alle macro, e quindi di strutturarle tenendo conto di questi argomenti. Inoltre, nelle definizioni delle macro si possono inserire direttive condizionali che consentono di scegliere di volta in volta quale blocco di codici dev'essere duplicato nel sorgente. Le possibilità quindi si moltiplicano a mano a mano che il preprocessore cresce di complessità.

Supponiamo per esempio di avere a che fare con un microprocessore dotato di un set d'istruzioni poco flessibile, mentre preferiremmo programmarne un altro dotato di un set d'istruzioni più interessante, ovviamente senza dover cambiare computer. Per farlo è sufficiente definire nel proprio ambiente di programmazione in Assembly delle macro istruzioni che emulino le istruzioni del microprocessore più evoluto. In pratica queste macro diventano il nuovo set d'istruzioni di cui ci serviamo. Questo semplice esempio, anche se più appropriato al caso di un 6502 che di un 68000, evidenzia la flessibilità delle macro istruzioni nel riassumere un intero blocco d'istruzioni con un unico nome simbolico.

Ovviamente, le macro istruzioni trovano la loro migliore applicazione quando determinati blocchi di codici ricorrono molto spesso in un programma.

Si tenga però presente che questi blocchi di codici ricorrenti devono essere abbastanza corti da non giustificare la loro trasformazione in routine.

Molto spesso, quando si parla di macro si adotta il linguaggio tipico delle routine, probabilmente perché in superficie le due cose, per quanto profondamente diverse, possono apparire simili. È facile per esempio sentir parlare di "chiamata alla macro", oppure leggere che "la macro esegue...". Si tratta di espressioni fuorvianti, che mal interpretano il concetto di macro. In questo articolo si dirà invece "l'inserimento di una macro nel sorgente...", "l'espansione della macro esegue..." e per indicare la trasformazione del nome di una macro nel relativo blocco di codici, diremo che la macro viene espansa dal preprocessore.

La notazione standard delle macro

Le macro usate nella programmazione dell'Amiga fanno uso di uno standard riconosciuto da diversi assembler, a partire da quello della Manx. Esaminiamo da vicino che cosa prescrive tale notazione.

Prima di tutto, i nomi di macro e di direttive assembler, per quanto non sia vincolante, vengono scritti in maiuscolo (CALLLIB, LINKLIB...). Le direttive più importanti nella definizione delle macro sono le seguenti: MACRO indica l'inizio di una macro; EQU assegna un'etichetta al valore di un'espressione; IFC confronta due stringhe di caratteri e se sono uguali impone all'assembler di continuare la compilazione con le successive linee di sorgente; IFNC confronta due stringhe di caratteri e se sono diverse impone all'assembler di continuare la compilazione con le successive linee di sorgente; IFGT elabora un'espressione numerica e se il risultato è maggiore di zero impone all'assembler di continuare la compilazione con le successive linee di sorgente; DS.B definisce uno spazio di memoria in byte; DS.W definisce uno spazio di memoria in word; ENDC conclude il blocco di linee sottoposte alla compilazione condizionale; ENDM indica la fine della definizione della macro; SET imposta una variabile o un'etichetta con il valore di un'espressione e XREF, infine, genera il file di riferimento incrociato (cross reference) relativo a tutte le variabili definite nel sorgente. Ricordiamo che le direttive condizionali consentono di escludere dalla compilazione determinate linee di sorgente a seconda del valore che assumono particolari espressioni nel corso della compilazione. Più precisamente, se una condizione non è verificata, una direttiva condizionale interrompe la compilazione fino alla prima direttiva ELSE o ENDC. Per esempio, le direttive condizionali vengono usate quando in un sorgente sono presenti linee da compilare soltanto se è stato indicato un particolare microprocessore, ma possono trovare largo uso in molte altre applicazioni, come nel debug o nei sorgenti che prevedono l'adattabilità dei codici a situazioni diverse.

Oltre a quelle elencate, riveste particolare importanza la direttiva FAIL. Quando l'assembler incontra FAIL, visualizza sullo schermo un messaggio definito dal programmatore e causa un errore di compilazione (si noti che sebbene questa direttiva in effetti generi un errore che impone all'assembler di non creare il file oggetto, non blocca la compilazione, e permette di controllare se si verificano altri errori). Questa direttiva è molto utile nella fase di definizione della macro per stabilire se è stata inclusa indicando un numero eccessivo di argomenti. Le macro che controllano il numero di argomenti indicati nella loro definizione, iniziano con la direttiva condizionale IFGT, la quale verifica che il numero di argomenti indicati non sia superiore a quello atteso ed eventualmente passa ad eseguire FAIL (si noti che con queste due direttive non viene invece rilevata l'eventuale mancanza di parametri nell'espansione della macro). Si vedano come esempi le macro CALLLIB e LINKLIB, tenendo conto che la variabile NARG è una variabile speciale dell'assembler che contiene sempre il numero di argomenti passati a una macro al momento della sua espansione.

Per convenzione, tutti i nomi relativi ai registri del microprocessore 68000 vengono indicati in maiuscolo, mentre le istruzioni mnemoniche previste dalla Motorola, come "jsr", "lea" o "move.l" sono indicate in minuscolo, sebbene gli assembler non impongano limitazioni.

I parametri formali

Come nella definizione delle funzioni (prevista dal linguaggio C), le macro Assembly possono ricevere argomenti formali: tuttavia, a differenza del C, una macro li usa al momento dell'espansione senza che siano stati dichiarati. Si tratta di argomenti che nella definizione di una macro vengono individuati tramite la loro posizione anziché tramite il nome. Questo significa che nella definizione di una macro è possibile per esempio riferirsi al terzo argomento specificando il numero 3 preceduto dalla barra inversa " \ " (i numeri che seguono la barra inversa possono variare da uno a nove). Se per via dell'assenza di nomi simbolici si teme di dimenticare il significato degli argomenti ai quali si fa riferimento, si può ricorrere a un commento da affiancare tramite il carattere "*" alla dichiarazione della macro (la prima riga della definizione). Se i riferimenti all'argomento della chiamata (\1, \2, \3...) sono racchiusi tra apici, l'argomento viene trattato come una stringa, altrimenti viene considerato una variabile numerica: in entrambi i casi il valore che l'argomento assume al momento dell'espansione viene sostituito a tutti i suoi riferimenti presenti nella definizione. Quando una macro viene espansa, i parametri formali presenti nella sua definizione vengono sostituiti dai corrispondenti parametri attuali indicati come argomenti dopo il nome della macro.

Le macro relative alle librerie

La Tavola 1 raccoglie le macro che analizzeremo in questo articolo, e le divide in tre categorie. La prima riguarda le macro – tanto quelle generiche quanto quelle dedicate ai dispositivi – create per abbreviare i nomi delle librerie shared (d'ora in poi, chiameremo le librerie shared semplicemente librerie, e useremo la definizione completa soltanto per le librerie linked). Queste macro hanno la funzione di facilitare i riferimenti alle librerie durante la scrittura di programmi che fanno uso delle routine OpenLibrary, CloseLibrary, OpenDevice e CloseDevice (si tenga presente che i dispositivi dell'Amiga non sono altro che librerie gestite in modo particolare, quindi parlando di librerie ci si riferisce anche ai dispositivi, a meno che non sia specificato diversamente). Inoltre, ogni programma che utilizza queste macro diventa di fatto immune da successive modifiche nei file INCLUDE perché, anche se le espansioni delle macro sono suscettibili di alterazioni, i loro nomi restano invariati.

Queste macro sono di ridottissime dimensioni.

Tavola 1

Nome della macro	Nome del file INCLUDE	Descrizione della macro
Macro che definiscono i nomi delle librerie		
AUDIONAME	devices/audio.i	Nome della libreria del dispositivo Audio
CIAANAME	hardware/cia.i	Nome della libreria della risorsa hardware CiaA
CIABNAME	hardware/cia.i	Nome della libreria della risorsa CiaB
DISKNAME	resources/disk.i	Nome della libreria della risorsa Disk
DOSNAME	libraries/dos.i	Nome della libreria DOS
EXECNAME	exec/execname.i	Nome della libreria Exec
EXPANSIONNAME	libraries/expansion.i	Nome della libreria Expansion
ICONNAME	workbench/icon.i	Nome della libreria Workbench
MISCNAME	resources/misc.i	Nome della libreria delle risorse generiche
PARALLELNAME	devices/parallel.i	Nome della libreria del dispositivo Parallel
POTGONAME	resources/potgo.i	Nome della libreria della risorsa Potgo
ROMBOOT_NAME	libraries/romboot_base.i	Nome della libreria RomBoot
SERIALNAME	devices/serial.i	Nome della libreria del dispositivo Serial
TD_NAME	devices/trackdisk.i	Nome della libreria del dispositivo TrackDisk
TIMERNAME	devices/timer.i	Nome della libreria del dispositivo Timer
Macro per la gestione delle librerie		
CALLLIB	exec/libraries.i	Macro per la chiamata di generiche funzioni di libreria
LINKLIB	exec/libraries.i	Macro utile per aprire una libreria e chiamare una funzione
LIBDEF	exec/libraries.i	Macro per la definizione degli offset ai vettori delle funzioni all'interno di una libreria
LIBINIT	exec/libraries.i	Macro per l'impostazione del primo offset in una tavola di vettori
CALLSYS	asmsupp.i	Richiama una funzione senza che sia necessario specificare i caratteri "_LVO" (fa uso di CALLLIB)
LINKSYS	asmsupp.i	Esegue il link a una libreria inserendo automaticamente i caratteri "_LVO" (utilizza LINKLIB)
LIBENT	libraries/dos_lib.i	Macro per la definizione degli offset dei vettori in una libreria DOS
EXTERN_LIB	exec/types.i	Macro per la definizione di un generico simbolo esterno
XLIB	asmsupp.i	Sinonimo della macro precedente
XREF_EXE	macros.i	Sinonimo della macro EXTERN_LIB, specifica per le librerie dell'Exec
XREF_GFX	macros.i	Sinonimo della macro EXTERN_LIB, specifica per le librerie grafiche
CALLEXE	macros.i	Macro analoga a CALLLIB, che include però i caratteri "_LVO"
LINKEXE	macros.i	Macro che effettua il link alla libreria Exec, utilizzando la macro LINKLIB
LINKGFX	macros.i	Analoga alla precedente, ma dedicata alla libreria Graphics
Macro per la gestione delle librerie dei dispositivi		
DEVINIT	exec/io.i	Macro analoga a LIBINIT, ma riferita a librerie di dispositivi
DEVCMD	exec/io.i	Macro utilizzata per aggiornare il contatore dei comandi presenti nella libreria di un dispositivo
BEGINIO	exec/io.i	Macro che manda in esecuzione la funzione BeginIO della libreria dei dispositivi, facendo uso di LINKLIB
ABORTIO	exec/io.i	Come sopra, per la funzione AbortIO

Consistono del semplice nome, per esempio AUDIONAME, e di una stringa di caratteri ASCII a terminazione nulla (cioè una stringa che termina con un byte a zero) che definisce il nome della libreria o del dispositivo. Un tipico esempio è:

```
AUDIONAME MACRO
        DC.B 'audio.device',0
ENDM
```

Alcune macro per le librerie utilizzano al posto della direttiva DC.B la macro STRING così definita (si veda anche il file INCLUDE exec/strings.i):

```
STRING MACRO
        DC.B \1
        DC.B 0
        CNOP 0,2
ENDM
```

Qui viene utilizzata la direttiva assembler CNOP per forzare la posizione dei caratteri della macro STRING a un indirizzo di memoria pari. Un esempio tipico di macro che impiega la macro STRING è:

```
PARALLELNAME MACRO
        STRING 'parallel.device',0
ENDM
```

Questo diverso modo di rappresentare una stessa macro (un fatto non certo positivo) mette in luce le variazioni subite dai file INCLUDE nel corso del tentativo, da parte dei programmatori della Commodore-Amiga, di utilizzare macro sempre più complesse per facilitare il proprio lavoro.

Le ultime due categorie di macro riportate nella Tavola 1 mostrano 18 macro dedicate alla gestione delle librerie e dei dispositivi. Molte sono esempi di macro che contengono al loro interno altre macro con argomenti predefiniti: per esempio, la macro CALLEXE contiene la macro CALLLIB e indica per essa particolari argomenti; la macro LINKEXE contiene LINKLIB e ne definisce anche l'unico argomento.

Spiegazione di alcune macro

Le macro relative alle librerie e ai dispositivi vengono usate soprattutto durante il link dei vari moduli oggetto che compongono il programma, al fine di risolvere tutti i riferimenti ancora irrisolti relativi alle funzioni delle librerie e dei dispositivi. Questi particolari riferimenti nei moduli oggetto sono gli offset ai vettori delle funzioni (delle librerie o dei dispositivi) che l'assembler non ha potuto risolvere, cioè inserire numericamente, perché sono specificati solo nelle librerie linked, alle quali l'assembler non accede.

La macro **CALLLIB** ha la seguente definizione:

```
CALLLIB MACRO * ftnVtrOstName
        IFGT NARG -1
        FAIL !!! too many arguments !!!
        ENDC
        jsr \1(A6)
ENDM
```

Questa macro, quando viene opportunamente espansa nel sorgente, serve a chiamare una funzione contenuta in una libreria, per esempio la funzione ClearScreen() della libreria graphics. library. La macro CALLLIB si aspetta un solo argomento: l'offset che – sommato all'indirizzo base della libreria – individua il vettore di salto alla particolare funzione indicata. Nel corso dell'espansione, viene eseguita la direttiva FAIL solo nel caso che vengano indicati due o più argomenti. Si noti che i caratteri che seguono la direttiva FAIL non intervengono nella compilazione, quando la direttiva viene eseguita, ma appaiono sullo schermo.

L'espansione della macro corrisponde all'istruzione `jsr offset(A6)`, la quale somma l'offset all'indirizzo base della libreria che i codici precedenti devono aver già memorizzato nel registro A6, e cede il controllo all'indirizzo assoluto che ottiene. Questo indirizzo individua il vettore di salto della libreria, che a sua volta cede il controllo alla funzione associata a quell'offset e a quell'indirizzo base. Si ricordi che per qualsiasi libreria (anche quella di un dispositivo), questo offset è un numero negativo, e che quindi i vettori di salto alle funzioni della libreria sono tutti distribuiti a indirizzi minori rispetto all'indirizzo base. Inoltre, si tenga presente che mentre l'offset viene determinato in fase di compilazione, il contenuto del registro A6 viene invece stabilito in fase di esecuzione dei codici: quando nel file eseguibile la CPU giunge all'istruzione `jsr offset(A6)`, i codici che precedono questa espansione di CALLLIB devono aver già memorizzato nel registro A6 l'indirizzo base della libreria, ottenuto chiamando OpenLibrary o OpenDevice. Per esempio, la struttura Library della libreria Exec è allocata in memoria a partire dalla locazione $00C00276. Se nel sorgente s'inserisce la macro CALLLIB seguita dall'argomento AllocMem, durante l'espansione la macro verrà trasformata nell'istruzione Assembly `jsr -198(A6)`. In fase di esecuzione del programma tutto funzionerà perfettamente se nel registro A6 viene inserito l'indirizzo base $00C00276 prima di giungere a questa istruzione.

La macro **LINKLIB** ha la seguente definizione:

```
LINKLIB MACRO * ftnVtrOstName, libraryBaseName
        IFGT NARG -2
        FAIL !!! too many arguments !!!
        ENDC
        move.l A6,(-SP)
        move.l \2,A6
        CALLLIB \1
        move.l (SP)+,A6
ENDM
```

Questa macro ha la funzione di generare i codici per aprire una libreria e chiamarne una funzione. Abbiamo dunque due argomenti: l'offset al vettore di salto della funzione, che è lo stesso argomento usato da CALLLIB, e un puntatore alla base della libreria che contiene la funzione. Si tratta di una macro analoga a quella precedente, ma che svolge un compito in più: generare in fase di compilazio-

ne anche i codici che inizializzano opportunamente il registro A6. Si noti che comunque il registro A6 viene aggiornato sempre durante il run-time, e non in fase di compilazione, dal momento che l'assembler non può sapere a quale indirizzo si troverà la base della libreria. Anche in questa macro, le prime due direttive controllano se gli argomenti passati sono più numerosi di quelli attesi.

La macro **LIBDEF** ha la seguente definizione:

```
LIBDEF MACRO * fntVtrOstName
        \1 EQU count_lib
        count_lib SET count_lib - lib_vectsize
ENDM
```

Questa macro serve a definire gli offset delle funzioni contenute in una libreria creata dall'utente. Riceve in input il nome di una funzione, poi, ogni volta che la macro viene eseguita, associa al nome della funzione un particolare offset (count_lib), calcolato riducendo il precedente della quantità lib_vectsize (si tratta di una costante pari a 6, il numero di byte occupato da ogni vettore di salto). Supponiamo che si debbano generare gli offset per una libreria contenente dieci funzioni, e che il vettore più vicino alla base della libreria debba corrispondere all'offset -30 (i quattro vettori precedenti, come nel caso delle librerie standard, li consideriamo riservati). Si inizializza allora count_lib a -30 tramite la macro LIBINIT (la prossima che analizziamo), e s'inseriscono nel sorgente dieci macro, una per ogni nome di funzione. Ovviamente, i nomi delle funzioni devono apparire come argomenti delle dieci macro: l'ordine sarà decrescente, dal nome relativo al vettore più vicino all'indirizzo base della libreria, per concludere con il nome relativo al vettore più lontano. La costante lib_vectsize viene definita insieme alla macro LIBDEF nel file INCLUDE exec/libraries.i.

La macro **LIBINIT** ha la seguente definizione:

```
LIBINIT MACRO * [vtrTableBaseOst]
        IFC '\1',''
        count_lib SET lib_userdef
        ENDC
        IFNC '\1',''
        count_lib SET \1
        ENDC
ENDM
```

Anche in questo caso si tratta di una macro di supporto alla creazione degli offset nelle librerie personalizzate, ovvero create dal programmatore. Ha la funzione di stabilire il valore della variabile count_lib, cioè il valore del primo offset utile da impiegare con la macro LIBDEF. Anche se si potrebbe pensare che il primo offset utile sia -6 (il primo vettore appena sotto l'indirizzo base), dobbiamo tenere conto che nella tavola dei vettori i primi quattro sono spesso considerati riservati, cioè individuano in memoria quattro funzioni riservate. Se si procede a creare una libreria che segua il modello delle librerie standard, si deve tenere presente che il primo vettore utile per le proprie funzioni è quello individuato dal consueto offset -30.

LIBINIT ha bisogno di un solo argomento, il nome dell'offset al primo vettore utile. La macro effettua un test su quest'argomento, attraverso la direttiva IFC, per controllare se si tratta di una stringa nulla. Se l'esito del confronto è positivo (nessun argomento), la macro associa alla variabile count_lib la costante lib_userdef (-30). Se invece è presente un argomento, la variabile count_lib assume quel valore. Analogamente a quanto visto prima, la costante lib_userdef è definita nel file exec/libraries.i.

Le macro **CALLSYS** e **CALLEXE** hanno la seguente definizione:

```
CALLSYS MACRO * ftnVtrOstName
        CALLLIB _LVO\1
ENDM

CALLEXE MACRO * ftnVtrOstName
        CALLLIB _LVO\1
ENDM
```

Queste due macro servono a richiamare una funzione contenuta in una libreria. L'unico argomento che si deve specificare è il nome dell'offset al vettore della funzione. Queste macro non contengono altro che la macro CALLLIB, e hanno soltanto lo scopo di evitare al programmatore la noia di digitare i caratteri "_LVO" (Load Vector Offset, l'offset che individua il vettore di salto alla routine della libreria).

Le macro **LINKSYS**, **LINKEXE** e **LINKGFX** hanno la seguente definizione:

```
LINKSYS MACRO * ftnVtrOstName, libraryBaseName
        LINKLIB _LVO\1,\2
ENDM

LINKEXE MACRO * ftnVtrOstName
        LINKLIB _LVO\1,_SysBase
ENDM

LINKGFX MACRO * ftnVtrOstName
        LINKLIB _LVO\1,_GfxBase
ENDM
```

Queste tre macro servono a semplificare ulteriormente il lavoro del programmatore nel chiamare le funzioni presenti in alcune librerie standard; fa eccezione la prima, che ha lo scopo di permettere l'impiego di una routine contenuta in una qualsiasi libreria (si devono specificare il nome del vettore e l'indirizzo base della libreria).

La macro LINKGFX permette di utilizzare le routine grafiche messe a disposizione dalla libreria grafica dell'Amiga: si noti che questa macro richiama al suo interno la macro LINKLIB, specificando l'offset del vettore e il nome della particolare libreria, ossia _GfxBase. Ancora una volta, l'impiego di questa macro evita al programmatore l'onere di digitare i caratteri "_LVO".

Lo stesso discorso si può fare per la macro LINKEXE, con una sola differenza: l'impiego della costante _SysBase, che punta alla base della libreria Exec.

La macro **LIBENT** ha la seguente definizione:

```
LIBENT MACRO * ftnVtrOstName
        _LVO\1 EQU count
        count SET count - vsize
ENDM
```

Questa macro serve a definire gli offset delle funzioni contenute in una libreria dell'Amiga. Si tratta di una macro molto simile a LIBDEF. L'unica differenza è che il nome della funzione indicata come argomento viene automaticamente preceduto dal prefisso _LVO, necessario per individuare le funzioni della libreria standard dell'Amiga. Nel file INCLUDE libraries/dos_lib.i la variabile count viene inizializzata a -30, mentre la costante vsize vale 6. In questo file INCLUDE, la macro LIBENT viene usata per definire gli offset di tutte le funzioni contenute nella libreria DOS.

Le macro **EXTERN_LIB**, **XREF_EXE**, **XREF_GFX** e **XLIB** hanno la seguente definizione:

```
EXTERN_LIB MACRO * ftnVtrOstName
        XREF _LVO\1
ENDM

XREF_EXE MACRO * ftnVtrOstName
        XREF _LVO\1
ENDM

XREF_GFX MACRO * ftnVtrOstName
        XREF _LVO\1
ENDM

XLIB MACRO * ftnVtrOstName
        XREF _LVO\1
ENDM
```

Queste macro sono tutte uguali e permettono di definire il nome di un indirizzo di funzione come simbolo globale esterno. In effetti, come si può notare, l'unica semplificazione che offrono queste macro è l'automatica aggiunta del prefisso _LVO ai nomi delle funzioni prima di dichiararle come simboli globali esterni. Analogamente al caso delle macro CALLSYS e LINKSYS, si tratta di macro che duplicano funzioni già espletate da altre macro dell'Amiga.

La macro **DEVINIT** ha la seguente definizione:

```
DEVINIT MACRO * [vtrTableBaseOst]
        IFC '\1',''
        cmd_count SET cmd_nonstd
        ENDC
        IFNC '\1',''
        cmd_count SET '\1'
        ENDC
ENDM
```

Questa è una macro di supporto alla creazione degli offset relativi alle funzioni contenute nelle librerie dei dispositivi. È molto simile alla più generica macro LIBINIT. Il suo scopo è impostare l'offset al primo vettore utile nella tavola dei vettori (come per tutte le librerie, questo offset si riferisce all'indirizzo base della libreria). Nel caso dei dispositivi, i primi vettori della tavola devono individuare alcune funzioni standard, cioè presenti in tutti i dispositivi. Seguono poi le funzioni non standard del particolare dispositivo.

Se non vengono dati argomenti, la macro imposta la variabile cmd_count al valore cmd_nonstd, che indica quante funzioni (o comandi) non standard sono presenti in quella particolare libreria; se invece è presente un argomento, questa variabile assume il suo valore. La macro e la variabile cmd_nonstd vengono definite nel file exec/io.i.

La macro **DEVCMD** ha la seguente definizione:

```
DEVCMD MACRO * cmdname
        \1 EQU cmd_count
        cmd_count SET cmd_count + 1
ENDM
```

Lo scopo di questa macro è contare tutti i comandi standard previsti da un particolare dispositivo. I comandi standard iniziano con CMD_RESET e terminano con CMD_FLUSH; si veda il file INCLUDE exec/io.i per comprendere meglio l'uso delle macro DEVINIT e DEVCMD. DEVCMD richiede come argomento soltanto il nome della funzione, al quale viene associato il valore del contatore cmd_count, che comincia subito a essere incrementato.

Le macro **BEGINIO** e **ABORTIO** hanno la seguente definizione:

```
BEGINIO MACRO
        LINKLIB dev_beginio, io_device(A1)
ENDM

ABORTIO MACRO
        LINKLIB dev_abortio, io_device(A1)
ENDM
```

Queste due macro sono molto simili. Servono per creare nel sorgente i codici che eseguono le chiamate rispettivamente alla funzione BeginIO o AbortIO di un particolare dispositivo, e contengono solo la macro LINKLIB seguita dall'offset della funzione e l'indirizzo base del dispositivo.

Prossimamente

Nel prossimo articolo della serie analizzeremo il pacchetto *CygnusEd Professional*, un editor di testi estremamente efficiente e veloce, che può essere usato per la stesura di programmi di qualsiasi tipo, indipendentemente dal linguaggio utilizzato, Basic, C o Assembly. Vedremo le possibilità di configurazione e i modi d'impiego di questo interessante prodotto.

Le capacità di *CygnusEd Professional* sono uniche. Permette infatti la gestione di diverse finestre con più file contemporaneamente aperti, e consente addirittura la visione di parti diverse dello stesso file in più finestre, il che si rivela utilissimo quando nel codice sorgente sono sparpagliate qua e là molte funzioni e molte routine. Inoltre, questo editor può interfacciarsi con altri programmi grazie a una serie di comandi che fanno uso delle possibilità offerte dall'*ARexx*, il famoso linguaggio batch originariamente pensato per i sistemi di computer IBM.

■

ESTRATTO DAL

CATALOGO LIBRI 1988/89

COLLANA TEMPUS scienza e tecnologia

La storia, l'impatto sociale e il futuro della scienza e dell'alta tecnologia

La IHT Gruppo Editoriale inaugura il suo terzo anno di vita dando l'avvio a una nuova collana che si occuperà di un settore ancora tutto da esplorare: quello della ricerca scientifica e tecnologica.

Con questa collana cominciano a precisarsi alcuni obiettivi della nostra casa editrice, che non intende limitarsi a pubblicare testi d'informatica – sia pure estremamente curati e scelti tra i più significativi del mercato mondiale – ma vuole offrire al pubblico un catalogo quanto più vario possibile che, privilegiando sempre il punto di vista tecnologico, possa aiutare i suoi lettori a tenersi aggiornati con ogni aspetto del mondo moderno.

Gli argomenti della collana, le sue tematiche, le sue linee d'interesse diventano sempre più evidenti leggendo la descrizione dei volumi presentati in questo catalogo. Non ci sono limiti "settoriali" alle esplorazioni di TEMPUS: vengono proposti testi sull'intelligenza artificiale, sull'inserimento dei nuovi sistemi "intelligenti" nella società di oggi, sui progressi compiuti nel campo della genetica, sull'evoluzione dell'economia moderna, sulle moderne teorie dell'evoluzione dell'universo, sui sistemi d'arma del futuro, sulla possibilità di essere ancora un "inventore" in un mondo in cui sembra esistere solo la ricerca di gruppo...

È una serie che si rivolge a tutti i lettori aperti ai nuovi stimoli della scienza, una scienza presente nella vita di ogni giorno, non solo nei laboratori e nelle università. E proprio per questo i libri che vengono selezionati per il marchio TEMPUS privilegiano la facilità di lettura, pur senza venire mai meno al rigore dei contenuti e alla completezza dell'informazione.

Noi non abbiamo mai creduto che solo chi è in possesso di un linguaggio matematico adeguato debba avere l'opportunità di affrontare il mondo della scienza. I linguaggi ermetici sono necessari soltanto per argomenti strettamente specialistici, e troppo spesso si leggono libri inutilmente ostici. Non abbiamo alcuna intenzione di battere ancora questa strada.

La collana TEMPUS propone libri stimolanti e vivaci, che possono aiutare l'uomo di oggi a entrare in contatto con la realtà affascinante in cui vivrà l'uomo di domani.

LA MACCHINA E LA MENTE

Alla scoperta della nuova scienza dell'intelligenza artificiale

George Johnson

È possibile creare una macchina che comprenda il linguaggio umano? Che faccia scoperte originali e sappia imparare dai propri errori? Quanto vicini siamo a risultati di questo tipo? Quanto lontani potremo spingerci?

La macchina e la mente offre un ritratto vivido e profondo di una scienza che sta nascendo, una relazione dalle più lontane frontiere a cui si sia spinto l'uomo: l'intelligenza artificiale (AI). È l'irresistibile storia di un'impresa che coinvolge intrallazzi politici, speculazioni selvagge, ostacoli impossibili, grandi trionfi e abbaglianti visioni.

In questo libro facciamo conoscenza con i padri dell'intelligenza artificiale: Marvin Minsky, Roger Schank, Douglas Hofstadter e Edward Feigenbaum. Dalle loro dirette esperienze scopriamo a che punto è arrivata la progettazione di una macchina che comprende il linguaggio umano e su quali teorie scientifiche è basata.

Accanto alla storia puramente tecnica dell'intelligenza artificiale, inoltre, *La macchina e la mente* prende in considerazione anche le implicazioni etiche e filosofiche di una linea di ricerca tanto delicata. Il cervello è solo un computer, che elabora una serie di programmi chiamati "mente"? Ha senso pensare di creare sistemi più intelligenti dei loro stessi creatori? È pericoloso lasciare che siano le macchine a prendere certe decisioni? Se daremo la capacità di ragionare alle macchine, daremo loro anche l'autocoscienza?

In 400 pagine, questo libro offre molte risposte, e moltissimi interrogativi, dal momento che oggi siamo arrivati al punto di poter parlare di "scienza dell'intelligenza artificiale" (e di raccontarne la storia), ma intravediamo un cammino ancora molto lungo. In un certo senso ci sentiamo sul confine tra il naturale e l'artificiale... ma non siamo del tutto certi che questo confine esista davvero.

400 pagine - 16 × 23,5 cm - rilegato con copertina rigida - ISBN 88-7803-012-0 - L. 42.000
Data di pubblicazione: ottobre/dicembre 1988

INVENTORI DEL NOSTRO TEMPO

Interviste con 16 famosi inventori americani

Kenneth A. Brown

Inventori del nostro tempo è una raccolta di 16 interviste con alcuni dei maggiori inventori di questo secolo. Da professionisti della ricerca come Maxime Faget della NASA, Harold Rosen della Hughes Aircraft e Bob Gundlach della Xerox, fino a imprenditori indipendenti come Stanford Ovshinsky e Raymond Kurzweil, esperto di intelligenza artificiale.

Le loro realizzazioni – il laser, il microprocessore, il pacemaker, il computer Apple II e molte altre – sono spesso coraggiose e originali, e sempre brillanti.

A poco a poco, attraverso le diverse storie dei personaggi intervistati, cominciamo a intuire quali regole comandano i meccanismi della dote che chiamiamo "creatività". Ma l'economia capitalistica rende la vita difficile allo spirito creativo, tanto che molti, già oggi, pensano che gli inventori siano una razza estinta.

Invece, come testimonia questo libro, si tratta di un'ipotesi assolutamente infondata... almeno per ora.

400 pagine - 16 × 23,5 cm - rilegato con copertina rigida
ISBN 88-7803-010-4 - L. 42.000
Data di pubblicazione: ottobre/dicembre 1988

COMPUTER IN GUERRA: FUNZIONERANNO?

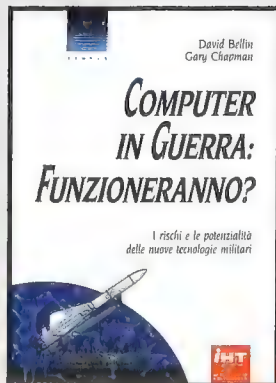
I rischi e le potenzialità delle nuove tecnologie militari

David Bellin e Gary Chapman

Stiamo per caso affidando la sicurezza del mondo a tecnologie troppo avveniristiche? Questa è la domanda a cui si sforza di rispondere *Computer in guerra: funzioneranno?*

Questo volume presenta gli elementi scientifici concreti necessari per crearsi un'opinione al riguardo, e riserva un ampio spazio alle analisi dei maggiori esperti. Il risultato è un libro non tecnico, di facile lettura e molto stimolante.

Tra i principali argomenti affrontati da *Computer in guerra: funzioneranno?* troviamo lo "scudo stellare", il programma "strategic computing", le applicazioni dell'intelligenza artificiale alla tecnologia militare e così via.

350 pagine - 16 × 23,5 cm - rilegato con copertina rigida
ISBN 88-7803-011-2 - L. 39.000
Data di pubblicazione: ottobre/dicembre 1988

I CREATORI DEL DOMANI

Un mondo nuovo di macchine organiche

Grant Fjermedal

«Siamo sulla soglia di un cambiamento paragonabile a quello che ha visto nascere la vita...». *I creatori del domani*, un libro brillante e originale, ci conduce in un viaggio straordinario verso le frontiere del futuro, fino ai limiti massimi a cui gli scienziati di oggi osano spingere la loro immaginazione.

Attraverso ampie interviste e approfonditi profili, l'autore ci fa entrare in un mondo che sarà il regno delle macchine pensanti. Provate a immaginare di trovarvi in una sala operatoria: un chirurgo robotico vi incide il cranio con delicatezza e mette a nudo il vostro cervello, ne registra l'intero contenuto e lo trasferisce all'interno di un computer... la vostra mente (qualcuno direbbe la vostra anima) si trova ora in un corpo artificiale, e il vostro vecchio corpo – ormai obsoleto – viene disconnesso. Ma nel nostro futuro remoto c'è ben altro: computer organici non più grandi di un granello di sabbia, braccia robotiche in grado di manipolare la struttura del DNA e...

300 pagine - 16 × 23,5 cm - rilegato con copertina rigida
ISBN 88-7803-013-9
Data di pubblicazione: gennaio/marzo 1989

IL SEGRETO DEL SUCCESSO

Investire nell'economia di oggi: le sfide, le opportunità, i pericoli

G. R. Funkhouser e R. R. Rothberg

Dall'inizio degli anni Settanta, gli imprenditori statunitensi hanno sperimentato di persona quale abisso separi la crescita prevista e la crescita effettiva. Questo libro esprime un singolare punto di vista: il generale rallentamento dello sviluppo economico negli Stati Uniti potrebbe non essere temporaneo.

L'idea della crescita aveva affascinato a tal punto gli economisti americani che nessuno aveva pensato di porsi due domande essenziali: «Quale tipo di crescita?» e «A beneficio di chi?».

Attraverso l'analisi di spettacolari fallimenti (Atari, Bendix) e di clamorosi fenomeni di rapidissimo sviluppo (DuPont, Procter&Gamble), *Il segreto del successo* analizza le differenze tra crescita sana e crescita forzata, e suggerisce le strategie migliori per investire in situazioni economiche diverse.

300 pagine - 16 × 23,5 cm - rilegato con copertina rigida
ISBN 88-7803-014-7 - Data di pubblicazione: gennaio/marzo '89

Distributore nazionale

Messaggerie Libri
Via Giulio Carcano, 32 - 20141 Milano
Tel. 02/8438141 - 8467341 - Telex 310672 MESSIT I

Filiali

BARI: Via Caduti del Lavoro, 2/A - 70126 Bari - tel. 080/483880/1 - province: Bari, Brindisi, Foggia, Lecce, Matera, Taranto - **BOLOGNA:** Via del Tuscolano, 3/5 - 40128 Bologna - tel. 051/324610-323768 - province: Bologna, Ancona, Ascoli Piceno, Ferrara, Forlì, Macerata, Mantova, Modena, Parma, Pesaro, Urbino, Ravenna, Reggio Emilia - **CAGLIARI:** Via del Commercio, 27 - 09100 Cagliari - tel. 070/287902 - province: Cagliari, Nuoro, Oristano, Sassari - **FIRENZE:** Via Francesco Baracca, 183 - 50127 Firenze - tel. 055/430223 - 432312 - province: Firenze, Arezzo, Grosseto, Livorno, Lucca, Massa Carrara, Perugia, Pisa, Pistoia, Siena - **GENOVA:** Via Gelasio Adamoli, 261/263 - 16141 Genova - tel. 010/869722-869769 - province: Genova, Imperia, La Spezia, Savona - **MILANO:** Via Volta, 13 - 20089 Rozzano - tel. 02/8240951/2/3 - 8258252 - province: Milano, Bergamo, Brescia, Como, Cremona, Novara, Pavia, Piacenza, Sondrio, Varese - **NAPOLI:** Via Nazionale delle Puglie, km. 36.150 - 80013 Casalnuovo - tel. 081/8423233 - 8423246 - province: Napoli, Avellino, Benevento, Campobasso, Caserta, Catanzaro, Cosenza, Isernia, Potenza, Salerno - **PADOVA:** Via Danieletti, 41 - 35100 Padova - tel. 049/609711 - province: Padova, Belluno, Bolzano, Gorizia, Pordenone, Rovigo, Trento, Treviso, Trieste, Udine, Venezia, Verona, Vicenza - **PALERMO:** Via Savonarola, 9 - 90135 Palermo - tel. 091/404289 - 403165 - province: Palermo, Agrigento, Caltanissetta, Catania, Enna, Messina, Ragusa, Reggio Calabria, Siracusa, Trapani - **ROMA:** Via Valtellina, 79 - 00151 Roma - tel. 06/538941/2 - 5313797 - province: Roma, Chieti, Frosinone, L'Aquila, Latina, Pescara, Rieti, Teramo, Terni, Viterbo - **TORINO:** Corso Peschiera, 321/0 - 10141 Torino - tel. 011/728073 - 726746 - province: Torino, Alessandria, Aosta, Asti, Cuneo, Vercelli.

Come ricevere i libri IHT

Librerie, cartolibrerie, grossisti di libri, computer shop, sono serviti dal nostro distributore nazionale Messaggerie Libri. Le richieste possono essere fatte per posta o telefonicamente alla filiale di competenza. I privati possono prenotare i libri IHT tramite l'apposito tagliando pubblicato sulla rivista Commodore Gazette, oppure ordinarli telefonicamente. L'indirizzo a cui rivolgersi è:

IHT Gruppo Editoriale
Via Monte Napoleone, 9 - 20121 Milano
Tel. 02/794181 - 799492 - 792612 - 794122
Telex 334261 IHT I - Fax 02/784021

I prezzi possono essere modificati senza preavviso. I pagamenti possono essere effettuati in contrassegno, con addebito delle relative spese postali, oppure anticipati a mezzo assegno o vaglia postale intestato alla IHT Gruppo Editoriale. In quest'ultimo caso, è necessario allegare all'ordine fotocopia della ricevuta, e telefonare per informarsi riguardo all'importo delle spese postali.

NORME PER I COLLABORATORI

Siete interessati a collaborare con la nostra casa editrice in qualità di autori, articolisti o traduttori?
In linea di massima prendiamo in considerazione ogni proposta, ma il nostro lavoro verrà semplificato se seguirete alcune semplici norme:

autori
– inviare curriculum accademico e professionale, e una nota biografica
– piano dell'opera proposta, con dettagliata descrizione della struttura e dei contenuti
– dimensioni approssimative dell'opera e data di consegna

articolisti
– inviare curriculum accademico e professionale, e una breve nota biografica
– breve descrizione dell'articolo, della serie di articoli, o del listato proposto
– dimensioni approssimative dell'articolo (in cartelle)
– eventuale disponibilità a una collaborazione continuativa
– nel caso di un listato, si ricordi che l'autore deve scrivere anche un articolo di presentazione

traduttori
– inviare curriculum accademico e professionale, e una breve nota biografica, specificando prima, seconda e (eventualmente) terza lingua
– specificare se la proposta riguarda la collana Informatica (traduzioni di tipo tecnico) o le collane Cinema e Tempus

Ogni proposta va inviata a:
IHT Gruppo Editoriale
Servizio Collaboratori
Via Monte Napoleone, 9 - 20121 Milano
Tel. 02/794181 - 799492 - 792612 - 794122

Un computer chiamato AMIGA

L'AMIGA

Come il Commodore Amiga ha cambiato il mondo dei computer, così il volume *L'Amiga* proietterà la vostra immaginazione lungo nuovi orizzonti di creatività. Preparatevi a ottenere stupefacenti risultati dal vostro computer: sofisticate immagini video, suoni e musica, sequenze animate da registrare su videocassetta, e molto di più. In questo volume troverete inoltre consigli utili per ottenere il meglio dall'Amiga Basic e da prodotti software eccezionali come Deluxe Paint, Deluxe Music e Deluxe Video.

416 pagine, oltre 100 illustrazioni, L. 60.000

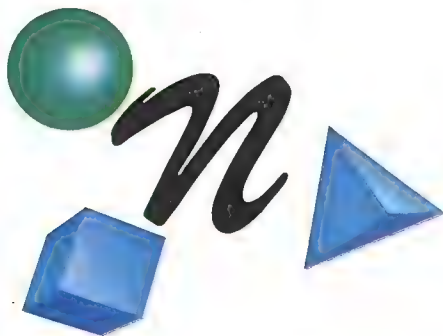
IL MANUALE DELL'AMIGADOS

Questo volume è l'unica documentazione ufficiale realizzata dalla Commodore sul sistema operativo dei computer Amiga. Contiene tre libri (Il Manuale per l'utente, Il Manuale per il programmatore e Il Manuale di riferimento tecnico) che costituiscono nel loro complesso la guida più completa per ogni utente dell'Amiga, dal principiante che vuole usare i comandi dell'AmigaDOS (che non sono spiegati dall'opuscolo in dotazione al computer), fino al programmatore evoluto che troverà utili informazioni per programmare in C e in Assembly.

376 pagine, L. 60.000

I libri IHT sono disponibili nelle migliori librerie e computer shop. Per ordini diretti servirsi dell'apposito modulo pubblicato a pagina 127
IHT Gruppo Editoriale - Via Monte Napoleone, 9 - 20121 Milano - Tel. 02/794181-794122 - Fax 784021 - Telex 334261 IHT I
Distribuzione: Messaggerie Libri - Via Giulio Carcano, 32 - 20141 Milano - Tel. 02/8438141

COMMODORE NEWS

NOVITÀ HARD E SOFTWARE DALL'ITALIA E DAL MONDO

ITALIA

EDILIZIA INFORMATICA

A cura del Dipartimento di Programmazione, Progettazione e Produzione Edilizia del Politecnico di Milano, a partire dal 5 maggio 1989, si svolgerà un corso di aggiornamento sul tema "L'informatica nelle imprese edili" (l'impatto organizzativo, le potenzialità e i metodi di governo delle nuove tecnologie).

Al corso possono essere ammessi dipendenti e collaboratori della Pubblica Amministrazione, operatori del settore edile, laureati in Architettura, Ingegneria, Informatica, Scienze Politiche, Economia e Commercio. Il corso tratteggia un quadro organico delle problematiche dell'informatica applicata all'impresa edile, fornendo un approfondimento conoscitivo sistematico e di ampia portata. Verrà dedicata particolare attenzione all'informatizzazione dell'impresa edile nel contesto attuale, con l'obiettivo d'individuare i criteri ottimali per utilizzare questa tecnologia sia in termini d'efficienza delle singole procedure, sia di controllo della gestione. Verranno esaminate anche le nuove frontiere applicative dell'informatica: reti locali, project management e intelligent building.

Per le domande d'ammissione e per ogni eventuale informazione rivolgersi al Politecnico di Milano.

Dipartimento di Programmazione, Progettazione e Produzione Edilizia
Via Bonardi, 3
20133 Milano
(tel. 02/23995100)

SOFTWARE ORIGINALE

In seguito a una riunione tenutasi a Firenze il 5 dicembre, i seguenti rivenditori di computer operanti in Toscana si sono impegnati a commercializzare solo ed esclusivamente software originale nei loro punti di vendita: Atema Sas, C.P.U. Srl, Help Computer, M.G. di Maurri, Teleinformatica Toscana, Punto Soft, Elettronica Centostelle.

Si tratta del primo impegno in Italia in tal senso e rappresenta una pietra miliare nel campo della rivendita del software, troppo spesso inquinata da copie pirata che oltre a svilire il prodotto, recano danni economici ai commercianti del settore. La riunione è stata promossa dalla Leader Distribuzione Srl di Casciago (VA), primaria ditta che importa e distribuisce in Italia giochi e software d'intrattenimento per computer.

La Leader ha dichiarato che investirà in Toscana almeno 25 milioni nei prossimi tre mesi per appoggiare questa decisione e sostenere i rivenditori che hanno sottoscritto l'impegno.

Nel congratularci con i rivenditori per questa dimostrazione di professionalità e serietà, ci auguriamo che presto questa iniziativa possa estendersi in altre zone d'Italia.

AMIGA TOTO

Amiga Toto è un programma di riduzione e condizionamento per sistemi integrali di Totocalcio. Offre (tra l'altro) la possibilità d'inserire colonne condizionanti, di ridurre le colonne del sistema in base alle probabilità percentuali di alcuni risultati, di ridurre il sistema in modo assoluto, d'impostare gruppi di due, tre e quattro segni riguardanti partite consecutive e definire in analogia con i risultati singoli presenza totale e

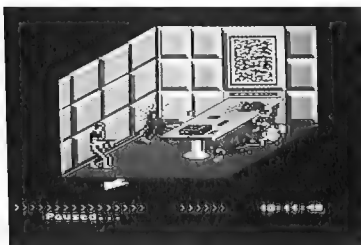
consecutività, di verificare il peso di una riduzione su un sistema integrale, di stampare il sistema su tabulato o su schedina con stampanti Epson compatibili e di rintracciare le colonne vincenti in un sistema già sviluppato.

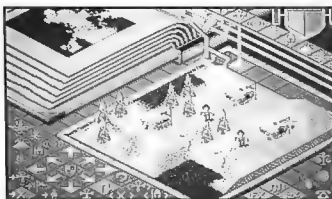
Amiga Toto è studiato per garantire un uso semplice e intuitivo, e offre la possibilità di ridurre il proprio sistema efficacemente e velocemente.

Leader Distribuzione
Via Mazzini, 15
21020 Casciago (VA)
(tel. 0332/212255)

ESTERO

ELECTRONIC ARTS NEWS

Project Firestart (C-64) è un gioco a metà tra l'horror e la fantascienza, dall'aspetto molto cinematografico: primi piani, dissolvenze e una colonna sonora particolarmente efficace. Il gioco

è ambientato nel 2066 e si svolge a bordo della *Prometheus*, una nave spaziale impegnata nella creazione, tramite sintesi genetica, di minatori in grado di lavorare nelle minere di titanio e iridio degli asteroidi. Dal momento che la prima generazione si è dimostrata piuttosto pericolosa, è stato necessario creare severi meccanismi di controllo. Ma dal 13 febbraio 2066 la *Prometheus* non dà più segni di vita: i sistemi di sicurezza non sono stati sufficienti? Il giocatore viene incaricato dagli Stati dei Sistemi Uniti di fare luce sul caso e parte alla volta dell'astronave per affrontare chissà quali orrori genetici.

Populous (Amiga) è una novità realizzata dai programmatori della Bullfrog, gli autori di *Fusion*. Si tratta di un gioco originalissimo, difficile da classificare, che vede due nazioni, una buona e una

malvagia, scontrarsi in nome del proprio dio. Il giocatore ricopre addirittura il ruolo della divinità, e il suo scopo è quello di accrescere la forza e il numero dei suoi fedeli, cancellando l'altro popolo dalla faccia del pianeta. Mostri marini e uccelli giganti agiscono come elementi di disturbo casuali, che attaccano senza distinzioni l'uno o l'altro popolo. *Populous* fornisce una mappa completa del mondo tridimensionale in cui si svolge il gioco e dà la possibilità di zoomare istantaneamente in ogni punto. Può essere giocato contro il computer o contro un altro Amiga.

Un'altra novità piuttosto inconsueta investe il settore librario: la Electronic Arts mette a disposizione dei giocatori più seri e tenaci una serie di *clue book* (libri d'indizi) che contengono suggerimenti, trucchi, mappe e percorsi per arrivare in un determinato punto del gioco o superare ostacoli particolarmente impegnativi. Finora sono disponibili i libri relativi a *Bards Tale I, Bards Tale II*, *Bards Tale III*, *Deathlord*, *Wasteland*, *The Mars Saga* e *Sentinel Worlds*, ciascuno al prezzo di 5 sterline. Gli appassionati non possono farseli sfuggire.

Electronic Arts
11-49 Station Road, Langley
Berkshire SL3 8YN, England
(tel. 0044/753/46672)

SHOOT'EM UP PIRATA

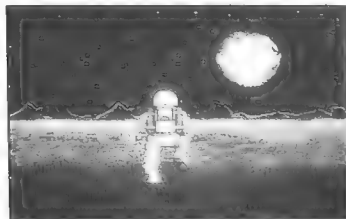
La data d'uscita della versione per l'Amiga di *Shoot'em up Construction Kit* è anticipata all'aprile 1989. La Outlaw ci informa che il programma è stato in buona parte riscritto e sono stati aggiunti molti miglioramenti.

Attenzione però alle copie pirata di questo programma che a quanto pare sono già ampiamente diffuse in Europa. Si tratta di copie di una pre-release, incompatibili con la versione ufficiale e incapaci di produrre giochi autonomi, e per di più contengono un virus particolarmente feroce e inattaccabile dalla maggior parte degli antivirus oggi disponibili.

Outlaw Productions
The Old Forge
7, Caledonian Road
London N1 9DX, England
(tel. 0044/1/2780751)

INFOGRAMES NEWS

Il popolarissimo personaggio a fumetti Tintin approda ora sulla Luna a bordo di... un Amiga. La Infogrames ha realizzato *Tintin on the Moon*, un gioco che trova Tintin e i suoi compagni di avven-

ture (tra cui gli inevitabili gemelli Thompson) precedere di altri quindici anni il primo uomo sulla luna. Il loro viaggio inizia nell'immaginaria Syldavia, anno 1953, ed è funestato da innumerevoli disavventure che vedono il nostro eroe alle prese con spolette da disinnescare, sabotaggi e rapimenti.

La stessa software house propone *Billiards*, una simulazione del biliardo francese per C-64 e Amiga da giocare contro il computer o contro un avversario umano, con tre opzioni per il tipo di stecca e la possibilità di vedere il tavolo in due dimensioni (dall'alto) o in prospettiva. Si possono regolare l'attrito con la sponda, la forza del rimbalzo, l'effetto e l'attrito con la superficie del tavolo.

Tra gli altri titoli proposti recentemente dalla Infogrames per l'Amiga segnaliamo *Purple Saturn Day* (giochi olimpici fantascientifici, nei quali si affrontano in quattro gare i rappresentanti di otto razze del lontano futuro), *The Temple of Flying Saucers* (un dopobomba ambientato nel terzo millennio, tra umani degenerati e superstiziosi, orde di mutanti e telepati uniti in una rete mentale che ricopre tutta la Terra), *The Quest for the Time-Bird* (versione computerizzata dell'avventura a fumetti *La cerca dell'augello del tempo* di Loisel e Letendre, pubblicata anche in Italia). Nella collana Infogrames/adventures in italiano è disponibile un altro gioco tratto dai comics: *I passeggeri del vento* (Amiga e C-64).

Infogrames
84, Rue de 1er Mars 1943
69628 Villeurbanne, Cedex, France

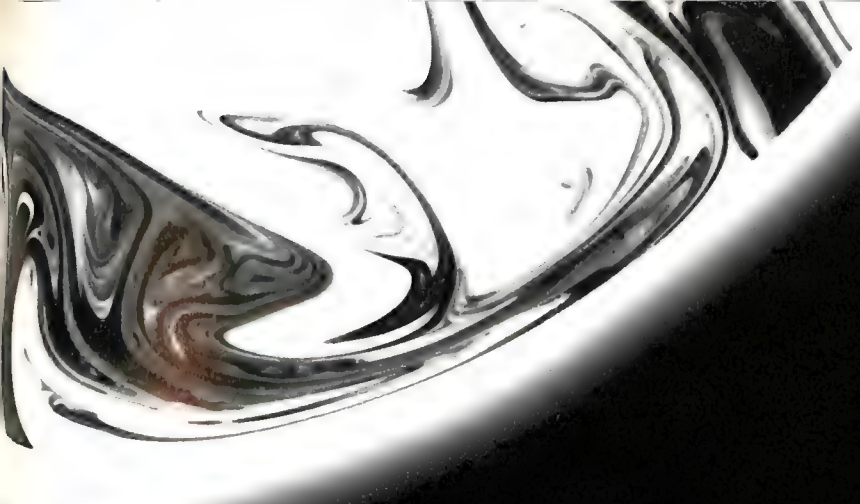
MICROPROSE NEWS

La Microprose, che da sei anni progetta e produce programmi d'intrattenimento per home e personal computer, è stata inserita nella lista INC 500, un elenco delle società private americane che hanno ottenuto i maggiori successi di crescita dal 1983 al 1987. Al 72° posto nel 1988, la Microprose è stata l'unica software house attiva nel settore dei giochi a comparire nella lista. Nel periodo indicato, le vendite della Microprose sono passate da 385 mila dollari a 13 milioni e 200 mila dollari, con un incremento del 3.332 per cento. La società spera di arrivare a cento milioni di dollari di fatturato per il 1990. «Penso che ci riusciremo» afferma Bill Stealey, presidente e cofondatore della Microprose, «abbiamo tutto quello che ci vuole: grandi persone con grandi idee».

E a proposito della Microprose, è recentissimo l'annuncio che la software house di Hunt Valley ha acquistato i diritti per tutto il mondo di *Universal Military Simulator II: Nation at War* (Amiga). Il programma sarà realizzato da un'équipe di programmatori capeggiata da Ezra Sidron (autore del primo *UMS*) e da Ed Bever (laureato in Storia a Princeton e supervisore della Microprose). Ezra Sidran ha dichiarato: «È sempre stato il mio sogno, poter lavorare in coppia con Bever. Lo considero uno dei maggiori esperti nel campo dei war game computerizzati, e sono particolarmente felice di svolgere questo lavoro con la Microprose, senza alcun dubbio la software house di maggior livello nel campo dei simulatori per home computer».

Microprose Software Ldt
2, Market Place, Tetbury
Glos GL8 8DA England
(tel. 0044/666/54331)

■

Manoscritti: le collaborazioni dei lettori – manoscritti, disegni e/o fotografie– sono benvenute e verranno valutate in vista di una possibile pubblicazione. Commodore Gazette non si assume comunque responsabilità per perdite o danni al materiale. Si prega di allegare una busta affrancata e indirizzata per ogni articolo. Il pagamento per materiale non richiesto viene effettuato solo in seguito all'accettazione da parte della redazione. I contributi editoriali (di qualunque forma) non si restituiscono. Tutta la corrispondenza editoriale, richieste di annunci, problemi di sottoscrizione abbonamenti, di diffusione e con gli inserzionisti, deve essere indirizzata a: Commodore Gazette - Uffici Editoriali - Via Monte Napoleone, 9 - 20121 Milano
La Commodore Gazette è un periodico indipendente non connesso in alcun modo con la Commodore Business Machines e con tutte le sue sussidiarie e affiliate, compresa la Commodore Italiana S.p.A. La Commodore Gazette viene pubblicata mensilmente dalla IHT Gruppo Editoriale, Via Monte Napoleone 9, 20121 Milano. Il costo degli abbonamenti è il seguente : Italia - 12 numeri L. 84.000, 24 numeri L. 168.000, 36 numeri L. 252.000. Estero - Europa L. 120.000 (10 numeri), Americhe, Asia... L. 180.000 (10 numeri). Nessuna parte di questa pubblicazione può essere in alcun modo riprodotta senza il permesso scritto dell'editore. La redazione si adopera per fornire la massima accuratezza negli articoli e nei listati pubblicati. La Commodore Gazette non si assume responsabilità per eventuali danni dovuti a errori od omissioni.

Qui Romaufficio a voi Managers.

FIERA DI ROMA
ORARIO 9,30-19,00
PROMOSSA
DALL'ISTITUTO MIDES

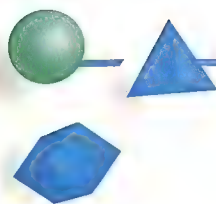
SERVIZIO INFORMAZIONI
A CURA
IBM ITALIA

FEDERLEASING
IN FIERA
A CANONI AGEVOLATI

ROMAUFFICIO '89

11ª MOSTRA DELLE NUOVE TECNOLOGIE
PER L'AZIENDA
LO STUDIO PROFESSIONALE
LA PUBBLICA AMMINISTRAZIONE

17-21 MARZO
5 giorni da non perdere

Classified

Software

Compro programmi gestionali per A2000 (possibilmente con manuali). Inviare offerte. Vendo CBM 64, drive 1541, *Speed-DOS*, 60/70 dischi pieni, *GEOS* originale, registratore, manuali ecc. Prezzo trattabile (vendo anche separatamente). Luca Ceria – Via Roma, 46 – 13060 Valdengo (VC).

Vendo 400 giochi per C-64/128 causa passaggio all'Amiga. L. 130.000 non trattabili. Francesco Cuzzola – Via Rucellai, 20/3 – 20126 Milano – Tel. 02/2550620.

Compro/scambio software per Amiga. M'interesso di tutto; per favore telefonatemi o scrivetemi. Luigi Gnocchi – Via G. Marconi, 17 – 21010 Mezzana di S. L. (VA) – Tel. 0331/768197.

Vendo programmi originali Accolade, SSI, Cinemaware, Activision (*Might & Magic*), simulazioni Electronic Arts (*Strike Fleet*), Sublogic (*Jet*), Actionsoft (*Up Periscope*), Microprose (*Gunship* e altri) e sportivi Epyx per C-64. Solo Roma. Luca 06/3765696.

Vendo Amiprint più 64 Emulator II per A500 a L. 50.000. Vendo dischi per C-64 pieni di giochi. Prezzo interessante L. 100.000 per 80 dischi più contenitore. I dischi sono per la maggior parte 3M. Davide Albertin – Via San Lorenzo, 58 – 15020 S. Giorgio M. (AL).

Scambio programmi, videogiochi e utilitiy. Posseggo molte novità per l'Amiga. Massima serietà e disponibilità. No lucro. Per informazioni telefonare allo 0572/771273. Chiedere di Gino.

Contatto utenti Amiga per scambio software (giochi, utility e demo) ed esperienze di ogni tipo. Per maggiori informazioni: Luca Castiglioni – Via Padre Ceriani, 19 – 20011 Corbetta (MI) – Tel. 02/9778142.

Vendo in blocco 50 dischi pieni di programmi per il C-64 a L. 50.000 (comprese eventuali spese di spedizione). Inoltre compro il programma *Laserterm* per il 6499 completo di manuali. Per maggiori informazioni scrivere a: Massimo Tabasso – Piazza Molineris, 1 – 12038 Savigliano (CN).

Scambio programmi per A-500 senza scopo di lucro. Non compro non vendo. Rispondo a tutti. Giuliano Mazzocchi – Via Pienza, 6 – 50142 Firenze.

Per Amiga scambio esperienze, programmi, utility. Preferibilmente zona Genova. Mauro Mele – Via Cardinale, 6/5 – 16035 Rapallo (GE) – Tel. 0185/274020. Oppure via modem su Opus Rapallo: 0185/274020.

Scambio programmi per C-128 sia in modo 128 sia in modo CP/M. Vasta biblioteca software con oltre 350 titoli, inviare vostre liste a: Rosario Albanese – C.so Alberto Amedeo, 66 – 90138 Palermo – Tel. 327764.

Cerco possessori Amiga per scambio programmi. Natale Bottoni – Via Sismondo, 1 – 54027 Pontremoli (MS) – Tel. 0187/830666.

Scambio programmi per Amiga 500. Tengo corrispondenza con gli interessati. Offresi e richiedesi massima serietà. Annuncio sempre valido. Massimo Falcione – Via Piedimontana – 84060 Omignano Scalo (SA) – Tel. 0974/836507.

Vendo programmi Amiga anche di produzione propria. Info-disk L. 5.000 in busta chiusa. Josef c/o Parrocchia Ognissanti – Via Appia Nuova, 244 – 00183 Roma – Tel. 06/7550697.

Scambio in Torino software per C-64. Cerco inoltre arretrati di *Zzap!* solo se in ottime condizioni. Telefonare a: Luca, 877016 Torino (20.00/22.00).

CLASSIFIED DELLA COMMODORE GAZETTE È UN MODO ECONOMICO PER INFORMARE LA PIÙ VASTA UTENZA COMMODORE SUI VOSTRI PRODOTTI O SERVIZI. GLI ANNUNCI NON A SCOPO DI LUCRO, INVIATI DA PRIVATI, VENGONO PUBBLICATI GRATUITAMENTE (COMPILARE L'APPOSITA SCHEDA DI SERVIZIO LETTORI).

Quote: 15.000 lire per linea, minimo 4 linee. Aggiungere 5.000 lire per ogni parola in grassetto o 50.000 lire per l'intero annuncio in grassetto.
Condizioni: pagamento anticipato. Vengono accettati assegni e vaglia postali. Gli assegni devono essere intestati a: IHT GRUPPO EDITORIALE s.r.l.
Forma: gli annunci sono soggetti all'approvazione dell'editore e devono essere scritti a macchina o in modo molto chiaro. Una linea equivale a 40 lettere, spazi tra le parole compresi. Pregasi sottolineare le parole che si intendono scrivere in grassetto.
Informazioni generali: gli inserzionisti devono sempre specificare nome e indirizzo completo. Gli annunci appariranno nel primo numero disponibile dopo il ricevimento.

Inviare il materiale a:
IHT GRUPPO EDITORIALE
UFFICI PUBBLICITARI
VIA MONTE NAPOLEONE 9
20121 MILANO

Attenzione: Commodore Gazette non si assume responsabilità in caso di reclami di qualnque natura da parte degli inserzionisti e/o dei lettori. Nessuna responsabilità è altresì accettata per errori e/o omissioni di qualsiasi tipo.

Con utenti 128/64 cambio/compro/vendo tutti i migliori programmi, completi di manuale originale o istruzioni in italiano. Massima serietà e competenza. Giuliano Cinci – Pian dei Mantellini, 44 – 53100 Siena – Tel. 0577/47054.

Cerchiamo nuovi contatti con utenti Amiga per scambio programmi, manuali, esperienze, impressioni e tutto ciò che riguarda questo computer. Offriamo inoltre lavori grafici di alta qualità e desideriamo contatti con programmatori assembler 68000. Estrema serietà. Sabaudian Software – Via Udine, 22 – 33032 Bertiolo (UD) – Tel. 0432/917520.

Scambio programmi per Amiga. Solo Milano e zone limitrofe. Annuncio sempre valido. Telefonare dopo le 20.00: 02/48840026 (Paolo).

Senza scopo di lucro scambio programmi per l'Amiga. Inviare propria lista. Annuncio sempre valido. Stefano Valduga – Via Giamaolle, 20 – 38051 Borgo Valsugana (TN).

Per C-64 scambio software di qualunque tipo. Dispongo di 1.300 programmi. Preferisco scambio su disco. Annuncio sempre valido. Walter Mughini – Via Z. Boccherini, 7 – 50144 Firenze.

Cedo programmi per Amiga sempre aggiornato con le ultime novità. No scopo di lucro. Chiedere lista. Scrivere a Lucio Lazzerini – Via S. Antonio, 49 – 38041 Abiano (TN). Telefono 0461/689294 (da lunedì a venerdì dopo le ore 22.00 – sabato dalle 12.00 alle 16.00 – domenica mattina).

Per Amiga 500 scambio games e utilities. Inviare lista a: Renato Romano – Corso Regina Margherita, 115 – 18014 Ospedaletti (IM).

Scambio programmi per Amiga soprattutto giochi. Solo in zona Pordenone. Telefonare dal lunedì al venerdì allo 0434/44944, ore pasti, chiedere di Davide.

Scambio programmi per Amiga 500 possibilmente in zona. Massima serietà. Rispondo a tutti. Scrivere o telefonare a: Roberto Ive – Via Don Gilardi, 8 – 10036 Settimo Torinese (TO) – Tel 011/8011527.

Hardware

Vendo Commodore 128D più monitor Philips BM7552, più registratore mod. C2N, più joystick, più cavo adattamento 80 colonne, più quattro libri Jackson – McGraw Hill a L. 1.000.000 trattabili. Regalo circa 25 cassette più 40 riviste. Fabrizio Frassa – Via Gorizia, 5 – 13040 Saluggia (VC) – Tel. 0161/486422.

Cedesi Atari 800 XL mai usato completo di registratore al miglior offerente. Rivolgersi a Giuseppe Cangiano – Via Mazzocchi, 133 – 81055 Santa Maria C.V. (CE).

Hardware vendo C-128 più drives 1541-71-81 exp. 640K, più registratore, più ROM upgrades 128 e 1571, più Epson LX 90, più monitor 1084, più programmi gestionali e manuali, più giochi e utilities programmazione Basic 7.0, più libri italiano inglese. Ottimo stato, imballi originali, prezzo concordabile. Paolo Sozzi – Via Arici, 21 – 25040 Monticelli Brusati (BS) – Tel. 030/6527449.

Vendesi C-128 perfettamente funzionante (modo 64-128-CP/M) più drive 1571 doppia faccia super speed, più registratore, più due cartridges "Final Cartridge II" e "Fast Load", più 50 dischi con i migliori programmi, più 20 cassette, più manuali. Tutto a L. 550.000. Diego Alberici – Via Ugo Foscolo – 27058 Voghera (PV) – 0383/649138.

Vendo Amiga 500 con il Kickstart 1.3, drive esterno, stampante Star NL 10 e archivio programmi. I programmi li posso scambiare con videoregistratore VHS recente. Silvano Bompieri – Strada dei Colli, 60 – 46040 Monzambano (MN) – Tel. 0376/800772.

Vendo Atari 1040 più mouse, più cento dischi, più cavo Scart a L. 1.000.000. Vendo scheda XT per A2000 più disk drive da 5,25" a L. 1.000.000. La scheda XT è in garanzia e ha un mese. Rivolgersi a Ezio al numero 031/550748 – Via Sinigaglia, 1 – 22100 Como.

Vendo stampante Commodore MPS 1200 usata pochissimo a L. 390.000. Telefonare allo 079/242019 ore pasti (Sandro, Sassari).

A sole L. 600.000 vendo un CBM C-128D (drive 1571 più C-128), più monitor fosfori verdi 40/80 colonne Philips, più datassette, più un joystick, più manuali di riferimento, più manuale di programmazione Basic, più giochi su disco e cassetta. Telefonare a Marco 06/3292768.

Occasione: vendo C-128 con copritastiera, due registratori, un play record (un anno di vita, imballi originali, manuali in italiano), più un libro di matematica e geometria per programmi Commodore, più 50 nuovi giochi con due portacassette per L. 450.000. Telefonare a Paolo 041/616725.

Occasione: vendo stampante STAR NL 10 in perfette condizioni! Cavo per collegamento con l'Amiga e qualsiasi MS-DOS compatibile più manuale. Telefonare allo 0578/294012 (Roberto) oppure allo 0575/67616 (Andrea).

Compro modem per Amiga standard V21, V22, (V23), in ottimo stato e a buon prezzo. Scambio software/hardware e manuali per l'Amiga 500. Diego Giorgi – Viale Cortemaggiore, 12/2 – 93012 Gela (CL) – Tel. 0933/938404.

Cerco disk drive 1541 per C-64 e stampante MPS 801/803. Spesa massima L. 150.000 cadauno. Sempre valido. Telefonare Pietro 19.00/21.00 – 011/6273665.

Vendo Amiga 1000 tastiera americana Kickstart (1.1, 1.2), monitor 1081, disk drive esterno 1010, il tutto perfettamente funzionante a L. 1.300.000. Antonio Durello – Via G. Durer, 31 – 35100 Padova – Tel. 049/612435.

Vendo espansione di memoria RAM 1700 da 128K per C-128 a L. 100.000. Francesco Di Maio – Viale Nettuno, 152 – 66023 Francavilla (CH) – Tel. 085/817520 ore pasti.

Vendo C-128, disk drive 1570, joystick, dischi e contenitore, manuali originali in italiano. Tutto a L. 600.000 trattabili. Telefonare allo 0141/291363 ore pasti e chiedere di Carlo.

Vendo a L. 500.000 C-64, disk drive 1541, registratore C2N, numerosi programmi su cassetta e disco, lezioni Beatrice D'Este nei contenitori in pelle. Christian Cognini – Via Petrarca, 68 – 60100 Ancona – Tel. 071/895244.

Vendo "Tre slot Motherboard" per C-64 a L. 25.000. Serve per collegare tre cartucce contemporaneamente, selezionabili tramite switch. Massima serietà. Paolo Sarego – Via del Terminillo, 51 – 02100 Rieti.

Varie

Cerco possessori di *Book of the Dead*, videogioco per il C-64, che conoscono le parole o le frasi per iniziare il gioco. Chi le conosce può mandarle per posta a Daniele Bagnasco – Corso Venezia, 40 – 14100 Asti. Grazie!

Cerco utenti di *GEOS 128* per scambio idee. Matteo Mairate – Via Fiume, 96 – 10088 Volpiano (TO).

Desidero contattare possessori di A2000 in zona Caldonazzo. Scrivere a Stefano Bernabè – Via Roma, 81 – 38052 Caldonazzo (TN).

Vendo anche separatamente circa 70 dischi con programmi, utility e giochi per C-64, registratore, joystick e copricomputer. Telefonare ore serali allo 0471/940876 o scrivere a Arturo Tonazzi – Via S. Giacomo, 131 – 39050 San Giacomo di Laives (BZ).

Vendo a L. 90.000 Enciclopedia De Agostini "Input": corso pratico di programmazione per lavorare e divertirsi col computer. Sei volumi, 1664 pagine già rilegati per Commodore 64. Paolo Demuti – Via San Protaso, 2 – 20026 Novate Mil. (MI) – Tel. 02/3543750.

Sono un neo-possessore di A2000 e cerco compagni di sventura per scambio impressioni, programmi, informazioni ed eventuale fondazione club in provincia di Belluno e Treviso. Andrea Tatto – Via La Cal, 47/D – 32020 Limana (BL).

Compro riviste inglesi e americane per Amiga (mi basterebbero anche solamente informazioni su di esse). Vendo inoltre libri per C-64/128 causa passaggio a sistema superiore. Maurizio Morini – Via Cosenza, 122 – 03100 Frosinone – Tel. 0775/200890.

Acquisterei anche in blocco i primi nove numeri di *Commodore Gazette*. Stefano Tirabassi – Via Ettore Rolli, 15 – 00153 Roma.

Se "smanettare" con l'Amiga è il tuo passatempo preferito, contact me! Ti aspettano: hot news, intro. personalizzate e tantissime altre cose... Fernando Carfagna – Via della Stazione, 27 – 04015 Priverno (LT) – Tel. 0773/911223. (The best get better...).

Commodore Club

È nato!! Cosa? Ma il "New Commodore Club", naturalmente! Chiamateci! Abbiamo grandi idee. Scrivete a New Commodore Club – Via F. Baracca, 9/11 A – 16035 Rapallo (GE) – Tel. 0185/60829.

Il "TNT" Club Alberobello cerca nuovi soci disposti a far esplodere il loro Amiga o MS-DOS con una miriade di programmi. Nessuna quota d'iscrizione, max serietà. Per informazioni telefonare a TNT Club – Via Monte Grappa, 2 – 70011 Alberobello (BA) – Tel. 080/721871 (chiedere di Giampiero).

Techsoft Club 64/Amiga! Non solo programmi, ma anche recensioni e notizie con un collegamento diretto tramite modem con una banca dati. Per informazioni telefonare o scrivere a Simone Prantero – Via della Pelliccia, 31 – Roma – Tel. 06/5890041.

Il Computer Club Vercelli esiste. Hai un computer (MS-DOS, Amiga, C-64, Spectrum o altro) e cerchi amici per scambiare idee, opinioni, programmi? Vieni a trovarci sabato pomeriggio in piazza C. Battisti, 7 (Ex-Enal). Poi deciderai se iscriverti. Per informazioni telefonare in ore serali allo 0161/63074.

8502 Group cerca soci per il club Amiga. Iscrizione gratuita: diritto al bollettino mensile. Luca Bassini – Via Pollone, 78 – 13051 Biella – Tel. 015/591273 (ore 20.00-21.00).

Cerchiamo soci per il nuovo club per C-64. Tutti gli iscritti riceveranno una rivista mensile con tutte le novità per il loro computer. Inoltre ogni nuovo socio ha diritto a 4 giochi in omaggio!!! Pierangelo Galizia – Via Appia S.S.7, 3 – 85050 Baragiano Sc. (PZ) – Tel. 0971/993258.

È sorto il "Tartugan Club Mazara" per Commodore Amiga e CBM 64. Per informazioni: TCM – Via Viareggio, 8 – 91026 Mazara Del Vallo (TP) – Tel. 0923/931302-945623.

INDICE DEGLI INSERZIONISTI

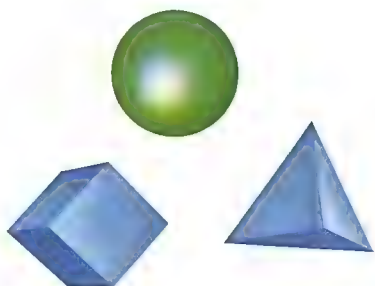
Direzione vendite spazi pubblicitari:

IHT Gruppo Editoriale
Commodore Gazette
Uffici Pubblicitari
Via Monte Napoleone, 9
20121 Milano
Tel. 02/794181 - 799492
792612 - 794122
Telex 334261 IHT I
Telefax 02/784021

Questo indice è da considerarsi come un servizio addizionale. L'Editore non si assume alcuna responsabilità per eventuali errori e/o omissioni.

Indirizzare eventuali lamentele riguardanti gli inserzionisti a:

Commodore Gazette
Uffici Pubblicitari
Relazioni Inserzionisti
Via Monte Napoleone, 9
20121 Milano

Nessuna responsabilità viene altresì assunta dalla Commodore Gazette per eventuali problemi di qualsiasi natura con gli inserzionisti. La responsabilità di quanto pubblicato negli spazi pubblicitari è esclusivamente del committente.

NEL PROSSIMO NUMERO

Marzo/Aprile 1989

- **C-64/128 -** Novità e curiosità, recensioni e listati: il C-64 e il C-128 non si scordano mai.
- **Il linguaggio dell'Amiga -** Impariamo a configurare il sistema in ambiente C e a sfruttare al meglio disk drive e RAM disk: terza parte del corso di C.
- **Hardware alla sbarra -** Terminata la serie dedicata alle stampanti, iniziano le nuove prove hardware di Commodore Gazette.
- **L'Amiga al microscopio -** Sotto esame alcuni indispensabili strumenti di programmazione: l'editor CygnusEd e i programmi Arexx e WShell.
- **GEOS e la Berkeley -** GEOS è un mondo che, con le sue sorprese sempre nuove, fa riscoprire il fascino del C-64.
- **Amiga BASIC -** La grafica da Basic: come gestire gli schermi dell'Amiga, con una serie di brevi esempi.

SERVIZIO LETTORI

Questa scheda è valida fino al 30 aprile 1989

A. Come giudica questo numero di Commodore Gazette?

- ☐ 1. Ottimo
- ☐ 2. Molto buono
- ☐ 3. Buono
- ☐ 4. Discreto
- ☐ 5. Sufficiente
- ☐ 4. Mediocre
- ☐ 6. Insufficiente

B. Quale(i) articolo(i) di questo numero ha apprezzato maggiormente?___

C. Quale(i) articolo(i) di questo numero giudica peggiore(i)?___

D. Quali argomenti dovrebbero essere trattati nei prossimi numeri di Commodore Gazette?___

E. Con quale aggettivo descriverebbe Commodore Gazette?___

F. Quante persone leggono la sua copia di Commodore Gazette?

- ☐ 1. Uno
- ☐ 2. Due
- ☐ 3. Tre
- ☐ 4. Quattro o più

G. Ha dei suggerimenti?___

H. Quale(i) computer utilizza?

- ☐ 1. C-64
- ☐ 2. C-128
- ☐ 3. C-128D
- ☐ 4. Amiga 500
- ☐ 5. Amiga 1000
- ☐ 6. Amiga 2000
- ☐ 7. PC 1
- ☐ 8. PC 10
- ☐ 9. PC 20
- ☐ 10. Altro (specificare)___

I. Quale(i) computer intende acquistare nel futuro?

- ☐ 1. C-64
- ☐ 2. C-128D
- ☐ 3. Amiga 500
- ☐ 4. Amiga 2000
- ☐ 5. PC 1
- ☐ 6. PC 10
- ☐ 7. PC 20
- ☐ 8. Altro (specificare)___

L. È un acquirente dei libri della IHT? Se sì come li giudica?___

M. Quali altre riviste (sia d'informatica che non) legge abitualmente?___

N. Indichi quali sono i suoi maggiori interessi

- ☐ 1. Videoregistrazione
- ☐ 2. Hi-Fi
- ☐ 3. Fotografia
- ☐ 4. Automobili
- ☐ 5. Sport
- ☐ 6. Viaggi

Nome ___
Cognome ___
Indirizzo ___
Città ___
Prov. ___ **C.a.p.** ___
Età ___ **Professione** ___

COMMODORE GAZETTE **febbraio 1989**

SCHEDA ORDINAZIONE LIBRI

Con il presente tagliando desidero ordinare i seguenti libri:

Collana Informatica
- ☐ L'Amiga (Michael Boom) .... L. 60.000
- ☐ Il Manuale dell'AmigaDOS (Commodore-Amiga) .... L. 60.000
- ☐ Programmare l'Amiga Vol. I (Eugene P. Mortimore) .... uscita: settembre '89
- ☐ Programmare l'Amiga Vol. II (Eugene P. Mortimore) .... uscita: maggio '89
- ☐ Guida ufficiale alla programmazione di GEOS (Berkeley Softworks) .... L. 64.000
- ☐ Flight Simulator Co-pilot (Charles Gulick) .... L. 30.000
- ☐ Volare con Flight Simulator (Charles Gulick) .... L. 45.000

Collana Cinema
- ☐ Le mille luci di Hollywood (David Chell) .... L. 42.000

Collana Tempus
- ☐ Inventori del nostro tempo (Kenneth A. Brown) .... L. 42.000

Pagherò in contrassegno al postino la somma di L.............................+ spese postali (L. 6.000 per volume)

Nome ___
Cognome ___
Indirizzo ___
Città ___
Prov. ___ **C.a.p.** ___ **Tel.** ___
Firma ___

COMMODORE GAZETTE **febbraio 1989**

☐ Desidero inserire gratuitamente un mio annuncio nella rubrica CLASSIFIED (solo per i privati e per gli annunci non a scopo di lucro).

☐ Desidero inserire un mio annuncio nella rubrica CLASSIFIED. Allego assegno bancario o circolare o postale oppure fotocopia della ricevuta del vaglia postale per un totale di L. ______________________ . Il mio codice fiscale o partita IVA (per le aziende) è il seguente:______________________.

Attenzione: perché un annuncio venga accettato è necessario che sia stato compilato anche il questionario presente sull'altro lato di questo tagliando.

TESTO:____

____

____

____

Inserire all'interno di una busta affrancata e spedire a:

Commodore Gazette
Servizio Lettori
Via Monte Napoleone, 9
20121 Milano

- - - - - ✂ -

Inserire all'interno di una busta affrancata e spedire a:

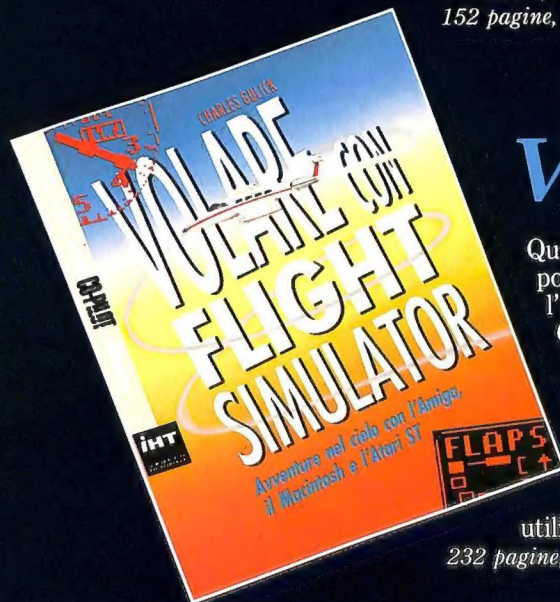
IHT Gruppo Editoriale
Divisione Libri
Via Monte Napoleone, 9
20121 Milano

UN COMPUTER, UN LIBRO...
...LE ALI

FLIGHT SIMULATOR CO-PILOT

Questo volume è un vero "istruttore di volo" per coloro che possiedono il programma Flight Simulator della Microsoft per i PC IBM e compatibili, oppure Flight Simulator II della SubLogic per Commodore 64, Apple II, Atari 800 XL e XE.

Non è un semplice compendio di comandi da ricordare a memoria, ma un brillante compagno d'avventura, scritto con estro e ironia. Un libro da sfogliare con il computer acceso, per imparare passo dopo passo i principi del volo "rettilineo e livellato", le procedure di decollo e di atterraggio, del volo strumentale e notturno, e del volo acrobatico.

152 pagine, L. 30.000

VOLARE CON FLIGHT SIMULATOR

Questo volume si rivolge a tutti gli utenti che possiedono la versione di Flight Simulator per l'Amiga, l'Atari ST o l'Apple Macintosh.

Con uno stile spigliato, ma con grande attenzione a ogni dettaglio, l'autore conduce l'allievo pilota attraverso le varie fasi di un corso di volo che si concluderà con un primo approccio al volo acrobatico. Il testo è completato da dettagliate cartine di rotta, schemi che illustrano l'uso dei comandi, parametri di volo ideali e procedure utili nelle più svariate situazioni di volo.

232 pagine, L. 45.000

I libri IHT sono disponibili nelle migliori librerie e computer shop. Per ordini diretti servirsi dell'apposito modulo pubblicato a pagina 127

IHT Gruppo Editoriale - Via Monte Napoleone, 9 - 20121 Milano - Tel. 02/794181-794122 - Fax 784021 - Telex 334261 IHT I

Distribuzione: Messaggerie Libri - Via Giulio Carcano, 32 - 20141 Milano - Tel. 02/8438141

REGALA IL CINEMA

LE MILLE LUCI DI HOLLYWOOD

Dietro le quinte del mondo del cinema

Ritorno al futuro
Guerre stellari
Voglia di tenerezza
Poltergeist

Urla del silenzio
Grease
Apocalypse now
Mishima
Gente comune
Witness, il testimone
E.T.

Brivido caldo
Amadeus
L'impero colpisce ancora
Star Trek
Rocky
Animal House
Silkwood

L'Esorcista
Il grande freddo
Provaci ancora, Sam
Gli intoccabili
Cocoon
Mission
Silverado

La scelta di Sophie
Tron
La febbre del sabato sera

IHT GRUPPO EDITORIALE

Interviste di David Chell

Un'inedita e affascinante raccolta d'interviste ai più famosi "creatori di cinema" degli anni '80. Questa volta non sono al centro dell'attenzione attori e registi, ma i tecnici. Tecnici premiati con l'Oscar, esperti in campi così particolari che molti di noi non ne sospettano neppure l'esistenza. Per curiosare dietro le quinte del mondo di celluloide. Per scoprire come si è trasformata la Hollywood degli anni '80. Per avere notizie sul futuro della computergrafica e sulle ultime novità nel campo degli effetti speciali. Ma anche per imparare ad amare il cinema e conoscerlo meglio.

I libri IHT sono disponibili nelle migliori librerie. Per ordini diretti servirsi dell'apposito modulo pubblicato a pagina 127
IHT Gruppo Editoriale - Via Monte Napoleone, 9 - 20121 Milano - Tel. 02/794181-794122 - Fax 784021 - Telex 334261 IHT I
Distribuzione: Messaggerie Libri - Via Giulio Carcano, 32 - 20141 Milano - Tel. 02/8438141